THE UNIVERSITY OF ALBERTA

STATISTICAL EVALUATION OF CP/67
A TIME-SHARING SYSTEM

by

(C) Arun K. Gatha

A THESIS

SUBMITTED TO THE FACULTY OF GRADUATE STUDIES

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE

OF MASTER OF SCIENCE

DEPARTMENT OF COMPUTING SCIENCE

EDMONTON, ALBERTA

Spring, 1970

THE UNIVERSITY OF ALBERTA

FACULTY OF GRADUATE STUDIES

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies for acceptance, a thesis entitled "Statistical Evaluation of CP/67, A Time-Sharing System," submitted by Arun K. Gatha in partial fulfillment of the requirements for the degree of Master of Science.

ABSTRACT

An empirical study of the time-sharing system CP/67 is reported in this thesis. Certain selected system and user parameters are studied to obtain system performance measures and an indication of the general behaviour of the users. A unique method for gathering data is given. The method involves making no change to the system internally but uses a virtual machine to obtain the required data.

The singularly important result obtained is the exponential relationship between the number of users and the paging load. The behaviour of the APL virtual machine in this time-sharing environment is studied in detail. The APL virtual machine is found to be demanding an extremely large amount of main storage. It is also found to be an "I/O bound" user.

The success of the "data-gathering" method indicates that a more detailed study of this sytem should be carried out. Suggestions for such studies are included in the thesis.

ACKNOWLEDGEMENTS

A study of this nature could not have been undertaken without the assistance of one who has considerable experience in the application of statistical techniques. I am particularly grateful to Professor U. M. von Maydell for her help, guidance, and invaluable suggestions as supervisor of this thesis.

I would also like thank Mr. G. Gabel and Mr. D. Webster for their constant interest and the hours of time they spent making this project feasible. Many thanks are due to the Department of Stenographic Services at The University of Calgary, and in particular to the typist, Mrs. M. van Ormondt.

Above all I would like to thank my wife, Anne, for her patience and constant encouragement to complete this thesis.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

NOMENCLATURES

RR	Round-Robin
FCFS	First-Come-First-Served
FB_2	Foreground-Background queueing model with two queues
FB_N	Foreground-Background queueing model with N queues
FB_∞	Foreground-Background queueing model with an infinite number of queues
CTSS	Compatible Time-Shared System
S.D.C.	System Development Corporation
I/O	Input/Output
CP/67	Control Program/67
CPU	Central Processing Unit
UTABLE	Primary User Control Table
ID	User Identification
CPRQUEST	Control Program Execution Request
Q_1	Queue one
Q_2	Queue two
PGWAIT	Page wait
SVC	Supervisor Call
CMS	Cambridge Monitor System
APL	A Programming Language
NUMUSERS	See Sec. 4.2
CPTIME	See Sec. 4.2
OVERHEAD	See Sec. 4.2

WAITTIME	See Sec. 4.2
PGREAD	See Sec. 4.2
PGSWAP	See Sec. 4.2
PROBTIME	See Sec. 4.2
TIMINQ	See Sec. 4.3
VMSTATUS	See Sec. 4.3
NOQUANT	See Sec. 4.3
CONQBIT	See Sec. 4.3
PGWT	See Sec. 4.3
IOWT	See Sec. 4.3
CFWT	See Sec. 4.3
EXCFN	See Sec. 4.3
TIMEUSED	See Sec. 4.4
NUMPAGES	See Sec. 4.4
PRIORIT	See Sec. 4.4
VTOTTIME	See Sec. 4.4
NUMDIAL	See Sec. 4.4
DIAG	Diagnose
LA	Load Address
CPTIME	See Sec. 5.1, page 50 for the definition of the transformed random variable
WAITTIME	See Sec. 5.1, page 50 for the definition of the transformed random variable
OVERHEAD	See Sec. 5.1, page 50 for the definition of the transformed random variable
PROBTIME	See Sec. 5.1, page 50 for the definition of the transformed random variable
PGREAD	See Sec. 5.1, page 50 for the definition of the transformed random variable

PGSWAP	See Sec. 5.1, page 50 for the definition of the transformed random variable
PGWT	See Sec. 5.1, page 50 for the definition of the transformed random variable
IOWT	See Sec. 5.1, page 50 for the definition of the transformed random variable
CFWT	See Sec. 5.1, page 50 for the definition of the transformed random variable
EXCFN	See Sec. 5.1, page 50 for the definition of the transformed random variable
Q1	See Sec. 5.1, page 50 for the definition of the transformed random variable
Q2	See Sec. 5.1, page 50 for the definition of the transformed random variable
EQ1	See Sec. 5.1, page 50 for the definition of the transformed random variable
EQ2	See Sec. 5.1, page 50 for the definition of the transformed random variable
OS/360	Operating System/360
PGLOAD	Paging Load
DOS	Disk Operating System

CHAPTER I

INTRODUCTION

Computer systems that are able to serve a single user in a conversional or interactive manner have been in existence for relatively many years. Lately, for economic and other reasons, interactive computer systems have been implemented to serve many users "simultaneously."

Directing attention to information processing techniques, such as data reduction, simulation models, casual "desk-calculator" computation, on-line experimentation, programming language translation, and data retrieval, it can be noticed that each of these processes puts its own specific requirements upon the hardware, software, and interface characteristics of the system in the sense of minimizing on cost, data and computation rates, storage sizes, and interface modes and languages. On-line experimentation, regardless of its specific nature, for example, implies sufficiently high data-rate sensors and internal processing speed to keep pace with the experiment, with appropriate display and input facilities for the user both to monitor and direct the course of the experiment. Data reduction and program language translation imply the ability to handle the input, output and internal processing. The rates and costs are determined by what the user is willing to accept. Simulation can cover a broad class of phenomena, some of which can place inordinate demands upon central processing speeds and costs. Data retrieval applications require a large amount of storage, ready access,

and comprehensible display of the data; while a personalized "desk-calculator" requires a minimum of an appropriate keyboard and printer with an easily usable programming language.

For any application, these processes are preceded by a period of program development and checkout, which may well require sizable amounts of computation. Thus, applications require the availability of low-cost computation capacity, which will handle both the debugging of the software and the production runs.

Sec. 1.1. Time-sharing Systems

The following three major types of computer systems are available. First there is the system which is usually "small" and relatively "inexpensive" and is operated by a single "on-line" user; that is, the computer is used through its manual input-output interface. Second, there is the "large," centralized "computer center" in which the user's programs are in batches and several may be executed at the same time by a multiprogramming operating system. Finally, there is the on-line "remote access" system with a distributed set of input-output terminals that can be connected to a centralized computing facility via public or private communication lines. To a user, such a system can be "on-line" in the sense that the processing equipment is capable of directly servicing his requests. This last category is usually called "time-sharing computing," since the processing techniques require time multiplexing of the users' requests.

At any given time in the operation of a time-sharing system, some

portion of the interactive users may require particular programs to be executed. The users' requests are selected in some given order, and each request is executed for a given time interval, not necessarily to completion. Typically, a particular user's program will be allowed to use a processing unit for a period of time, will be stopped so that another user's program can run, and at some later time will be continued from the point where it was stopped. At the point in time when one user's program is stopped and another's resumed, the status of the former must be saved and that of the latter restored, in order to continue a program from where it was discontinued previously.

This relatively new approach requires some justification and hence systematic analysis is necessary.

Sec. 1.2. Analysis and Measurements

Measurements made on time-shared systems have the following purposes:

1. to make a choice between one system and another, using the different possible configurations as decision criterion
2. to deepen the understanding of a system so as to optimize a given system or to affect the design of other systems.

Scherr (23), at the conclusion of his monograph states the following:

> With the advent of new time-sharing techniques and systems, much more analysis and measurements must be accomplished if these systems are to be designed and used intelligently. It is clear that no small effort should be expended in order to monitor and measure the performance and use of any operating time-shared system. It is through

intelligent use of this type of data that improvements can be made to present systems and that proposed systems can be evaluated.

The following three approaches are open to an analyst faced with evaluation of system's performance of a time-sharing system.

1. Mathematical Modelling. This involves modelling of time and space scheduling algorithms of the system. In most cases it is constrained to more or less simplified queueing models of these algorithms. This almost inevitable simplification of the physical system, so that for example "Markov-type" analysis can be applied, is one of the major drawbacks to an analyst using the mathematical approach. In spite of the above-mentioned difficulty, this is still the most widely used method of analysing time-sharing systems.

2. Simulation Models. Simulation models are very useful since the level of detail necessary in handling some of the features of time-sharing systems, are beyond the scope of mathematically tractable models. Markov models cannot, in general, be used to represent processes where other than random queueing is used. Queueing models are not applicable for processes where the arrival rates of service requests are a function of the service rate. This kind of problem can be easily overcome by simulation studies. They can be used in future prediction of the system performance by varying certain parameters like the size of storage and the speed of data transfer through a channel device. The detracting factor of a simulation model may be that it requires a large amount of care and computing time in order to incorporate the details necessary to represent a complex time-sharing system accurately.

3. Statistic Gathering. The method, in this final approach available to an analyst, and perhaps the closest to the real situation, involves gathering data on certain selected system parameters and user characteristics. We will discuss this last method and consider its advantages in the next section.

Sec. 1.3. Statistical Approach

The last of the three methods discussed in the preceding section has some distinct advantages over the other two methods. Comparing it with mathematical modelling and simulation it can be noticed immediately that it is not constrained with any of the necessary and simplifying assumptions binding the other two techniques. There are no *a priori* assumptions made. The method of "stat-gathering" allows us to arrive at conclusions based solely on the performance of the system as it behaved at the time of gathering this information. For example, the persistent assumption of Poisson arrival rate, made in most of the mathematical queueing models, is discarded. Instead, the actual "arrival times" are obtained through direct observation, which could then be used as the input to the other two models. Thus, the statistical approach is easier and more reliable than the mathematical and simulation approach.

A stat-gathering system or model should be such that it does not require too much storage space and processing time because this would imply biasing the data on the use of the system by the analyst's computations. The analyst should make full use of the fast-access storage available to him as a user of the time-sharing system under study.

The problem of data-reduction can also be a very detracting factor for the analyst using stat-gathering methods. This could be avoided effectively by carefully choosing the parameters to observe. The aim should be to obtain maximum information from minimum data. This involves a great amount of analysis completed prior to actually commencing with the data-gathering. This analysis could involve decisions like, how often should the observations be taken. The balance between the quantity of data and its usefulness should always be kept in mind. Gathering data for little or no additional information should be avoided. This would only put an unnecessary load on the time-sharing system and would probably be detrimental to its performance.

These three approaches are discussed in greater detail in the next chapter pointing out the amount of research carried out in modelling time-sharing systems.

CHAPTER II

REVIEW OF LITERATURE

In the late fifties the idea of "time-sharing" was first proposed by a group of computer-oriented scientists at M.I.T. and since it materialized in Project MAC and other implementations, research in such systems has been growing steadily. Research has been carried out extensively in the three main approaches of analyzing such systems.

In the next three sections, a review of the mathematical approach, the approach through simulation, and the empirical data-gathering (measuring) approach, respectively, is given.

Sec. 2.1. Mathematical Approach

Before proceeding with a review of the mathematical modelling of time-sharing system, some definitions are necessary. The definitions in this section pertain to the papers reviewed.

Time-sharing systems work in the following manner. Incoming tasks are queued and scheduled for service according to a particular scheduling policy. At its scheduled "service-time," each job is processed for a time period called a "quantum." If during this quantum the task is completed, that task departs and service begins on the next; otherwise, the uncompleted task rejoins a queue to await further service. The time the computer spends on scheduling, allocating, buffering, and controlling terminal input and output represents a slice of processing time that may be called "overhead." If not all the active parts of user-programs fit into main storage, processing time may be lost while awaiting the

interchange of the task just completing its quantum of service and the task that is next scheduled to run. This is usually referred to as the "swapping" time in the literature.

Mathematical models of time-sharing systems quite naturally are stochastic and their analysis thus draws heavily from queueing theory. A number of such models now exist; in general these differ only in the choices for the following factors:

1. The number of input queues can either be finite or can be considered to tend to infinity.

2. The number of central processors considered has not exceeded two.

3. The type of arrival processes used are Bernoulli, single Poisson stream, or multiple Poisson streams.

4. The type of service processes applied have been Geometric, Exponential, Erlang, or general.

5. The swapping time, say T_s, and the overhead time, say T_o, are either both considered to be zero or their sum, $T_s + T_o$, is considered to be a constant or a random variable.

6. The quantum length is either finite or tends to zero.

7. Some of the service disciplines used are the round-robin, the first-come-first-served to completion, and priority systems.

The main performance measurements considered are the following:

1. Average response time--the average time between the entering

of a request at the terminal and the acknowledgement that the request has been completed.

2. Average waiting time in the queue--The average time a request spends in the queue before receiving service.

3. Average queue length--The average number of requests in a queue.

A model of a time-sharing system with one processor, a finite number of channels, and round-robin scheduling (Fig. 2.1) has been studied extensively by Adiri and Avi-Itzhak (1), Greenberger (15), Scherr (23), Coffman and Krishnamoorthi (6), Krishnamoorthi and Wood (18), Patel (22), Krishnamoorthi (19), and Coffman (7).

Infinite input source model analysis was initiated by Kleinrock (16), who considered a round-robin scheduling policy again. Kleinrock (17), in a later paper also investigated the processor-shared model in which there are P priority groups. The input to the p^{th} priority group is Poisson, with average rate λp, and each arrival to the p^{th} group has an exponentially distributed service requirement with a mean of $1/up$ operations where up is the service rate and $p = 1, 2, \ldots, P$. A member of the p^{th} group is given $g_p Q$ seconds of service each time he reenters the queue where Q is the fixed quantum length and g_p a weight factor for the p^{th} group. Kleinrock in this paper studied the effect of the priority assignment on the total time a request spent in the system. He found that for each priority group there exists a critical value of the required service time below which the wait time in the RR (round-robin) system was less than in the "first-come-first-served" to completion system.

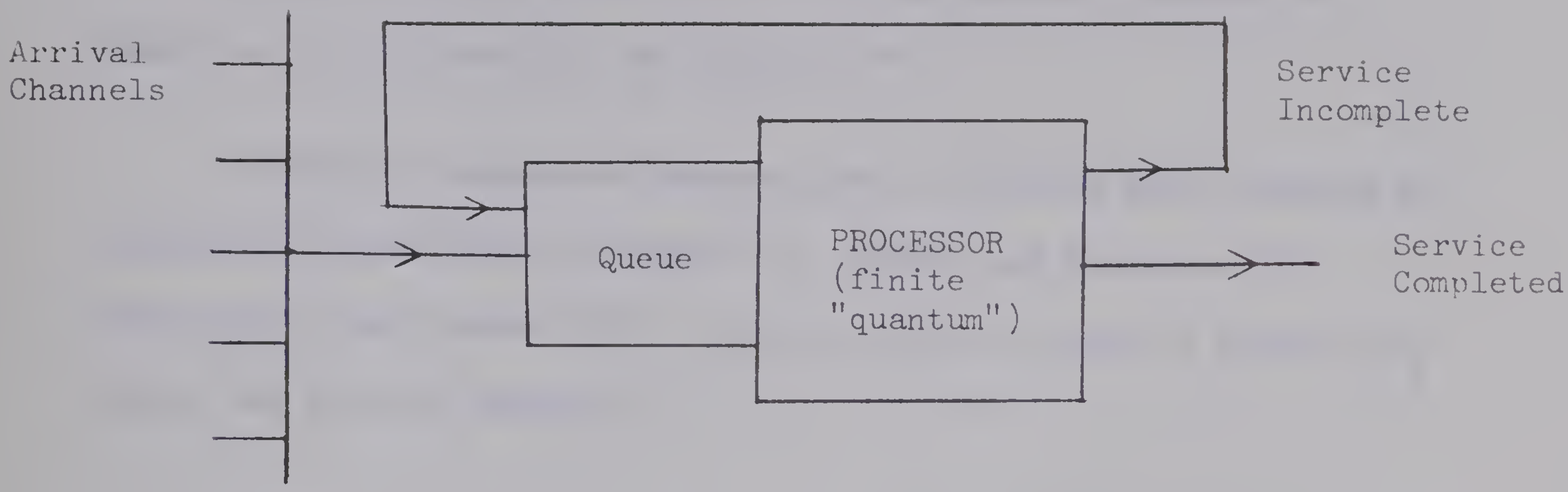

Fig. 2.1. Round-Robin (RR) Model

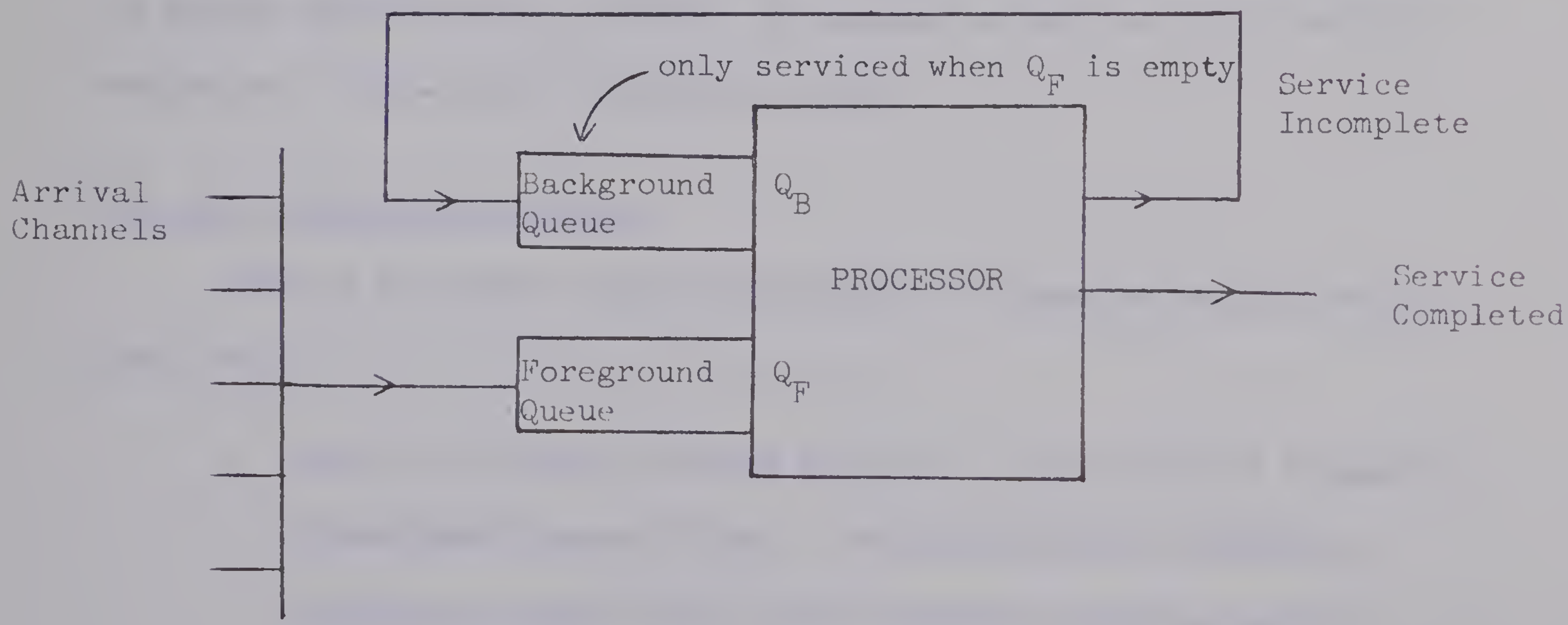

Fig. 2.2. Two-Level Foreground-Background (FB_2) Model

Others who have studied the infinite source RR system are Chang (4, 5), Coffman (8), and Shemer (27).

Foreground-background systems (Fig. 2.2) have been proposed and studied by Corbató (10), Coffman (7), Coffman and Kleinrock (9), Shemer (27), and Schrage (24). Studies have been made of general FB_N models and also FB_∞ models.

Another major scheduling policy to note is the "shortest remaining processing time" scheduling discipline which has been studied by Schrage and Miller (25) and Schrage (26).

As mentioned in Chapter I, all these papers make some assumptions on arrival and service time distributions to use queueing theory techniques. This is a drawback, but this type of analysis should provide the system designer with a number of degrees of latitude with which to synthesize a time-shared processing system.

Sec. 2.2. Simulation Models

Five of the major simulation efforts of time-sharing systems are given below:

1. Scherr (23) has simulated the M.I.T. Project MAC's Compatible Time-Shared System (CTSS). He considers the following simulation models; CTSS, CTSS with RR scheduling policy instead of the existing FB9 policy at Project MAC, and CTSS with multiprogramming to provide overlapped processing and swapping;

2. Fine and McIsaac (14) have simulated a system similar to the System Development Corporation (S.D.C.) Time-Sharing System. They consider a round-robin system as well as a two-queue, FB_2, scheduling policy;

3. Fife and Rosenberg (13) have simulated a system in which the central notion is one of memory-sharing with fixed memory partitioning. The main core is divided into five memory blocks. Each user is confined to remain within a single block. Jobs queue up for an allocation to core memory, and once in core, remain there until the job is completed or until the program halts, awaiting an operator message (or until an illegal command is generated). Processing is transferred to a new job in core when the current program being run fills its output buffer;

4. Lavita (20) has simulated a FCFS run-to-completion system with emphasis on the comparison of "response time" between a drum system and a disk system;

5. Neilson (21) has simulated Stanford's 360/67 time-sharing system. He varied the basic hardware (e.g. 1 or 2 processors), speed of data transfer to and from storage devices (e.g. disks) and obtained an optimum configuration of the desired system.

He also found that the capabilities of disks are far behind the capabilities of the rest of the system, considering the delay caused by disk I/O. He also investigated the difference between "paging" from disk and drum.

Sec. 2.3. Measurement Studies

There have been four major measurement studies of actual time-sharing systems.

1. Scherr (23) has made measurements of the behaviour and system performance of the M.I.T. Project MAC's CTSS (mentioned in Sec. 2.2.). During the period of his measurements, CTSS served approximately 250 users with an IBM 7094 augmented by a 7320 drum and a 130/-2 disk storage device. He obtained probability distributions of interarrival times (think time), service time, program size, and investigated relations between response time/processor time and number of users; computer utilization and number of users; and utilization of swap storage and number of users.

2. Totschek (29) reports measurements made on the S.D.C. Time-Sharing System which can handle up to 35 users at remote consoles. He obtained the probability distributions of think time, service time, program size, and number of users. Totschek also investigated (like Scherr) the relationship between (response time/processor time) and number of users, and tape usage and number of users.

3. Sutherland (28) has made measurements on the Lawrence Radiation

Laboratory Time-Shared System which is implemented on the CDC 6600. The system handles 48 teletype-equipped consoles which serve 110 users. He obtained relations between the assigned priority and program size; percentage of time spent on user programs and day, user, idle, and disk time and day, and finally, memory map and program size.

4. Cantrell (3) has reported his measurement of the GE-Dartmonth Time-Shared System. This system handles 200 users at remote terminals. He obtained the probability distributions of service time, program size and disk requests. He also investigated the relationship between breakdown of computer utilization and processor time.

Of primary interest with respect to user characteristics is the distribution of think times. Scherr finds a fair degree of agreement between simulation and actual measurements, and it is interesting to note that both yield an average think time of approximately 35 seconds.

The other major user characteristic is the distribution of service time (swap time excluded). Totschek comments that the log-normal distribution fits both service time and think time reasonably well. Cantrell observes that 90% of all service times represent just slightly more than 10% of the system load (as far as processor time is concerned) and, therefore, 10% accounts for almost 90% of the load.

The distribution of program sizes provides little agreement among

the above-listed sources. Scherr reports an average program size of 6.3×10^3 words. It is interesting to note at this point that all conclusions drawn from results obtained are based on "average" values. This could be a major setback.

The results obtained by Scherr for the ratio of response time to processor time as a function of mean number of interactive users for the CTSS measured data, the CTSS simulation and the RR simulation all agree remarkably well. Furthermore, they all agree with his mathematical, RR processor-shared model! Totschek's curves are similar in shape to that of Scherr's, but an absolute comparison is difficult to make because Totschek includes I/O time in the response time.

The other measurements concerned with system performance are widely dispersed in character and of a probing nature rather than strongly relevant to some model. They reflect the observations of Estrin, et al. (11) that present methods of measurements do not allow sufficient freedom in design of experiments on complex systems. Moreover, the documentation on actual measurements is poor in the sense that the method of obtaining these data is rarely given in detail.

In the next two chapters, a description of CP/67, the description of the statistical (empirical) evaluation model and the method of obtaining data is given.

CHAPTER III

A DESCRIPTION OF THE CP/67 TIME-SHARING SYSTEM

The description of the time-sharing system - CP/67, (Control Program/67), is given here.

Sec. 3.1. CP/67 - A General Description

CP/67 is a control program designed for execution on an IBM System/360, Model 67. Its objective is to create an environment in which each user believes that he has the complete resources of a System/360 Model 65 and can perform his own work under the supervision of the programming system of his choice. It achieves its objective by generating a "virtual computer" for each user and by sharing the resources of the real computer (e.g. CPU time, main storage) among the virtual computers for all users that are concurrently keyed into the system.

Virtual computers can be defined as computers which function like real ones but which are the products of software simulation. When a user identifies himself from a terminal, the control program "creates" for his personal use a virtual computer of a predefined configuration. The systems administrator defines this configuration of each user's virtual machine before the system becomes available to the users. This configuration can be different for different users. To the user, his virtual computer appears real and he uses it as if it were. The control program also provides commands that parallel the functions of switches on an operator's console. The user issues these commands from his

terminal, and, thus, the terminal becomes a pseudo-console for his virtual machine.

After the control program has created the virtual computer, the user equips it with the programming system that gives him the desired functional capabilities. The user--not the system--determines the facilities available to him.

The minimum hardware configuration required by CP/67 is given below.

1. 1403 Printer;
2. 2067-1 or 2067-2 CPU;
3. Two 2311 Disk Drives or 2314 Storage Unit;
4. 2365 Core Storage Unit;
5. 2540 or 2501 Card Reader;
6. 2540 Card Punch;
7. 2702(3) Transmission Control Unit;
8. 1052 On-line Console Typewriter.

The following optional devices are also supported:

1. 1051/1052 Data Communications System;
2. 2250 Graphic Display Unit;
3. 2820/2301 Drum Storage Controller(s)/Unit(s);
4. 2400 series Magnetic Tape Units;
5. 2741 (-1, -2) Communications Terminals.

Sec. 3.2. CP/67 - Time-Sharing Environment

CP/67 creates the time-sharing environment by:

1. Scheduling and allocating main storage space, CPU time, and I/O devices to the virtual computers.

2. Handling all interruptions (see Sec. 3.6.).

3. Protecting system files, user programs, and user data during execution.

The control program shares execution time in the central processing unit (CPU) among the virtual computers on a demand basis and on a scheduled basis. It schedules and allots units of CPU time, in rotation, to the virtual computers. When a particular virtual computer has used up its unit of time ("time slice"), the control program locates the next "runnable" virtual computer and passes control to it for a corresponding interval of time. If the virtual computer currently in control must wait for some event (an I/O operation), the control program gives control to another virtual computer, which has demanded CPU time. The detailed description of the module that carries out these tasks is given in Sec. 3.3.

The control program also uses a special technique to share main storage among concurrent users. This technique is called "paging." The objective of this technique is to keep in main storage only those portions of each user's program that are required at that time. This necessitates the segmentation of each user's program into manageable units. The units used in this case are 4096-byte blocks called "pages." By breaking programs into pages, main storage can be allocated in page increments and pages can be loaded dynamically for execution. Thus, at execution time,

main storage holds only the active part of each user's program. This paging technique is described in greater detail in Sec. 3.4.

The control program must also handle all the interruptions and change of state in the CPU necessitated by program or machine errors, completion of I/O operations and other conditions. Interrupt handling by CP/67 is described in Sec. 3.5.

Sec. 3.3. CP/67 - Dispatcher

The allotting of CPU time to different users and scheduling user-program execution is handled by the control program routine called DISPATCH. When all interruption-handling routines complete their processing, they transfer control to DISPATCH. This, then, is the central routine of the system CP/67. In general it performs two functions; first, it charges time used within the control program to the appropriate user, and second, it determines which user is to receive control next.

DISPATCH initially receives control after the system generation on each day and control program initialization. It remains idle until an interrupt occurs (i.e. a user signs on). The appropriate interrupt handler will log the user on and return control to DISPATCH. Logging a user on includes the following:

1. Allocating and initializing the primary user control table (UTABLE);
2. Checking user ID and password;
3. Allocating tables and reserving direct access storage space for paging; and,

4. Creating virtual I/O blocks to describe the user's virtual machine and chaining virtual device blocks to real device blocks.

There is one UTABLE for each user in the system. It is the primary control block from which all user blocks are strung. It completely reflects, along with the virtual I/O blocks, the status of the corresponding virtual machine. A description of the UTABLE is given in Appendix A.

Each time DISPATCH is entered, the time used by the current (interrupted) user within the control program is computed and added to the TIMEUSED entry in the user's UTABLE. If the current user has not exhausted his allotted time for this quantum (time slice), he will be eligible for restart. In this case his CPRQUEST's are honoured, his pending interrupts are reflected and then if not in "wait" state (i.e. runnable) he is restarted. If no time remains for the interrupted user, the next runnable user is selected.

DISPATCH, upon each entry to it, and prior to the running of any user, checks the queues of control program execution requests (CPRQUEST) for any pending work. If any requests are found, the appropriate execution request block (CPEXBLOK) is used to load the registers and dispatch control to a specified section of the control program. This section, on completion, returns control to DISPATCH. If the current user is not runnable and if all CPRQUEST stacks are empty, then a new user is selected to run.

In order to prevent paging overload, the system allows only a subset

of the users to run at any given time. Interactive users (needing little compute time) are in Q_1 and the users who put a heavy load on the system in terms of CPU cycles required or amount of nonterminal I/O done, are in Q_2. There is a maximum limit on both Q_1 and Q_2, which is set by system administrators and partially dependent upon the real core size of the computer.

A user is in one of the following five states at any time:

1. In Q_1; - State A
2. Waiting to get into Q_1; - State B
3. In Q_2; - State C
4. Waiting to get into Q_2 - State D; and
5. Dormant, not requiring system resources; - State E.

In addition, a user may or may not be runnable, regardless of whether he is in the queues. A user is not runnable if he is waiting for:

1. A page to be brought in;
2. An I/O operation; and,
3. A CP console function.

In order to select a new user, the first preference is given to a runnable user in Q_1. If no such user is found and Q_1 is not full and a runnable user is waiting to get into Q_1, then that user is selected. If no such user is found, then a runnable user will be selected from Q_2. If none exists, and Q_2 is not full, a runnable user waiting to get into Q_2 will be selected. If still no user is found, then CP goes into WAIT state.

To start (or restart) a user, DISPATCH loads the appropriate control registers from the contents of the chosen user's UTABLE entries, loads the interval timer with the user's quantum (0.05 seconds), or the unused portion of it, and gives control to the user by entering the problem mode.

The management of Q_1 and Q_2 is as follows: A user enters State A (in Q_1) only from State B (waiting to get into Q_1), and whenever Q_1 is not full. A user enters State B when he has a read operation on his terminal. When in Q_1, users are allowed to use up to 0.4 seconds of accumulated CPU time. One condition to remain in Q_1 is never to use a full time slice (quantum) or 0.05 seconds at any one time. If a user uses a full time slice without any console function, or uses 0.4 seconds in bursts of less than 0.05 seconds, he enters State D (waiting to get into Q_2). A user enters State C (in Q_2) from State D when Q_2 is not full. When in Q_2, users are allowed to use 5 seconds of CPU time, again in bursts of 0.05 seconds. A user enters State D again if he uses a full time slice in one burst, or uses an accumulated 5 seconds in bursts of less than 0.05 seconds. When there are many users in State D, the selection to enter State C is according to the number of time slices used by different users. A user, who has used the least number of time slices, is selected since he has the highest priority. (All times given in this paragraph are adjustable system constants and are the values used at The University of Alberta.)

Sec. 3.4. CP/67 - Paging

The paging technique employed by CP to share the main storage is discussed here.

When a user starts his session, the control program places the first page of the user's programming system into main storage. The page is loaded into an available block of main storage that starts on a page boundary. The page is not necessarily loaded at the same relative main storage position as it would occupy were the programming system running on a real computer.

As the user's program is executed, the hardware dynamically converts references to relative (virtual) addresses into actual main storage addresses. When the program refers to an address in a page that is not in main storage, an interruption occurs and the control program loads the required page into main storage. While this is being done, the current user (say user A) is put in "PGWAIT" and another user is selected to run. When user A is selected to run at a later instant, execution continues with the referenced addresses being dynamically relocated.

Because of this dynamic address relocation feature, the pages of a user's program need not occupy contiguous locations and may be scattered throughout main storage. Also, because of the high demand for main storage in a multiple-user environment, the control program shares main storage among the active pages of the programming systems of competing users.

Finally, when main storage is completely filled, and it becomes necessary to bring in another page, page swapping occurs. An appropriate page of a user's program in main storage is transferred to secondary storage (e.g. disk or drum) and the required page is brought into main

storage in its place. If a page to be replaced has previously been swapped, and has not been modified, since it was last swapped, it is not necessary to write it onto secondary storage because a copy already exists there. When the particular page that was replaced is required, it is obtained from secondary storage and swapped with one that is in main storage.

Sec. 3.5. CP/67 - Interrupt Handling

Before proceeding with the interruption-handling by CP, it is necessary to define the three possible states of the real computer. When instructions in the control program are being executed, the real computer is in the supervisor state; at all other times it is in the problem state, unless, of course, no user is runnable; in this case the computer is in "idle" state.

The interruption handling routines determine the cause of the interrupt by checking the machine state. The reason is to charge time to a user for the particular interrupt, if the machine (real computer) is found to be in problem state. If the machine is found to be in supervisor state, then no user is charged for interrupt handling. An example of the latter is the hardware timer interrupt every 0.05 seconds.

There are four major types of interruptions handled by CP.

1. Supervisor Call (SVC) interrupts;
2. External interrupts - These include timer interruption;
3. Program interruptions - These are caused by paging interrupts and privileged instructions;

4. I/O interruptions - These are caused by I/O requested by a user or by CP itself.

In the next section, the description of the CP/67 system, as it exists at The University of Alberta Computing Centre, is given.

Sec. 3.6. CP/67 - The University of Alberta Configuration (June, 1969)

The CP/67 system at this campus operates for four hours a day on the IBM Series 360, model 67 computer. A diagram representing 360/67 is given in Fig. 3.1.

The two main programming systems available, and used, are CMS (Cambridge Monitor System) and APL (A Programming Language). The operating system, OS/360 has been tried under CP.

CMS is the monitor that creates the conversational part of the time-sharing environment. CMS could be used without CP/67 to create a conversational system without time-sharing capabilities. When jointly used, the monitor executes under the supervision of the control program. 360/Assembler, and FORTRAN IV are available through CMS.

APL is implemented in a very special way. One virtual machine defined as a System/360 Model 65 is equipped with the Disk Operating System (DOS) which supports the APL/360 time-sharing system. Thus, one time-sharing is supported by another time-sharing system. Usually about half of the terminal users are using the APL virtual machine. APL machine has the special scheduling priority so that it remains in Q_1.

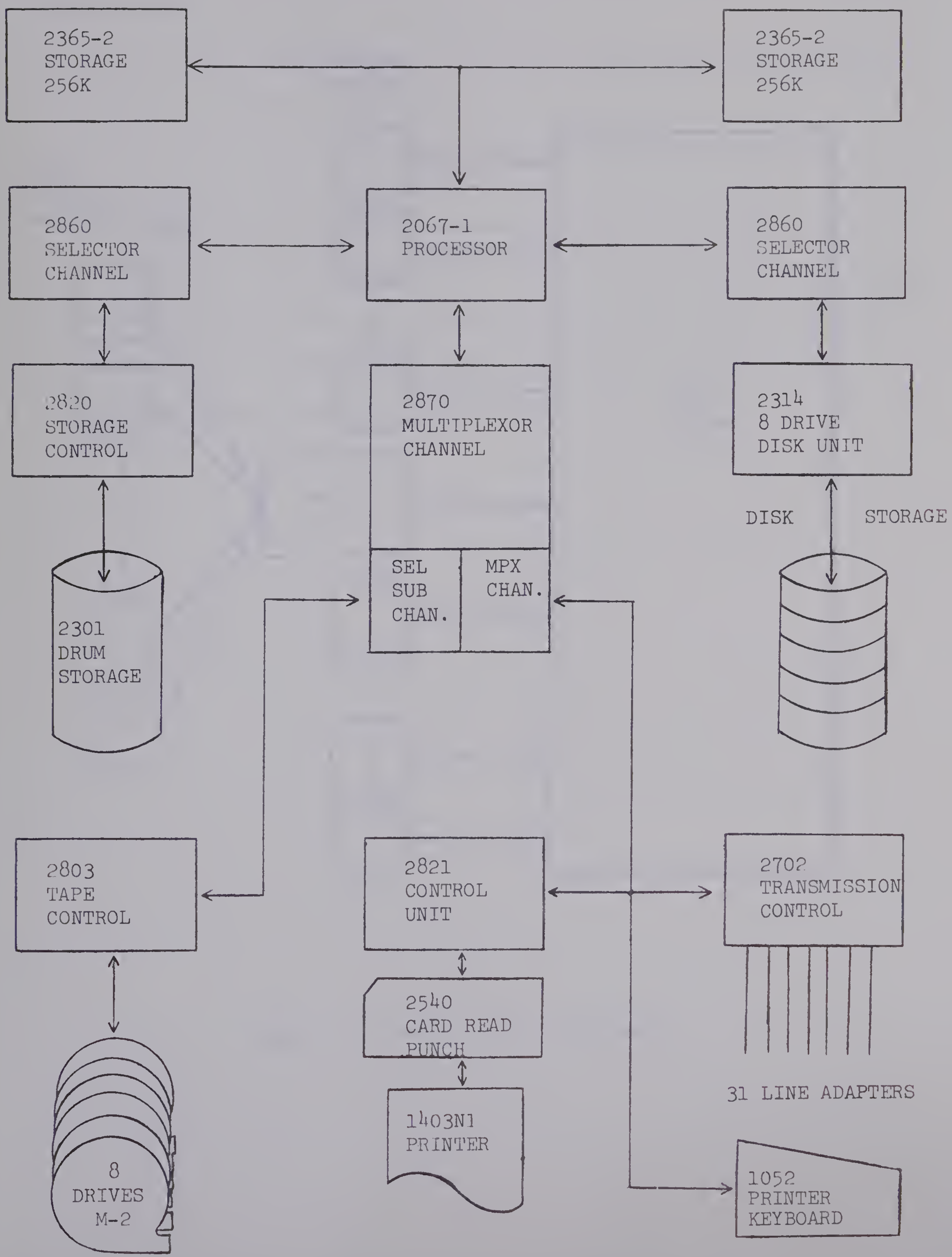

Fig. 3.1. 360/67 Hardware Configuration

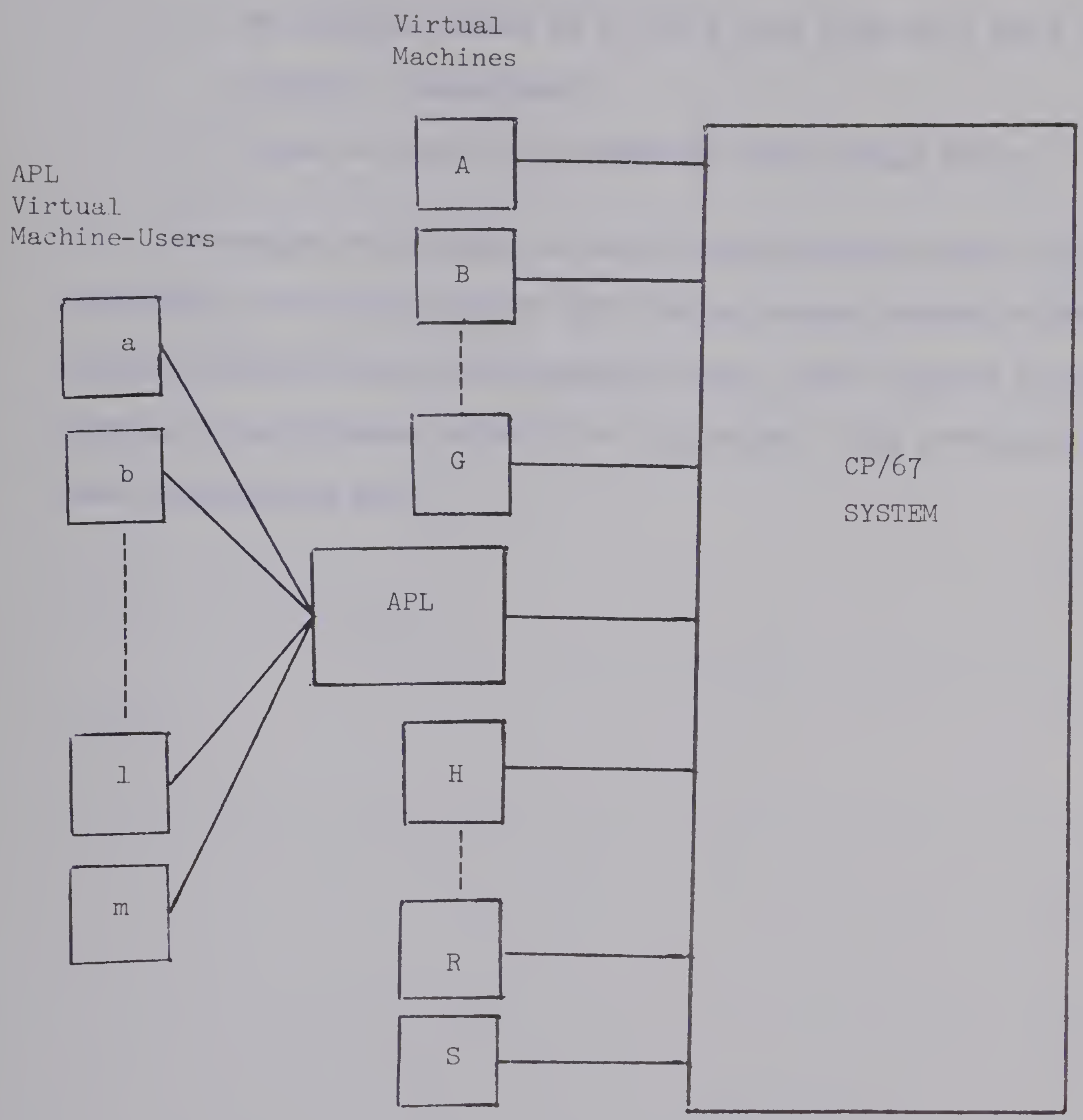

Fig. 3.2. APL in CP/67 Environment

Two other points to remember are:

1. The maximum lengths of Q_1 and Q_2 are fixed at 9 and 6 user requests, respectively.
2. Paging is carried out from 2314 Disk Storage Device.

To analyze this system, a statistical evaluation model was constructed. Care was taken to avoid using extreme amounts of processing time and available core while gathering data. This resulted in careful choosing of performance criteria for the system. This particular model is described next.

CHAPTER IV

EVALUATION MODELS

The considerations, that need to be accounted for, in the statistical evaluation of any system, have been mentioned previously in Sec. 1.3. and also at the end of the last chapter. The most cumbersome problem to consider is that of data reduction. The stress, again, is put on obtaining maximum information from minimum amount of data without making too many restricting assumptions.

Sec. 4.1. The Three Models

The three models used to evaluate CP/67 statistically are the following:

1. System evaluation--this requires data on system parameters; for example, the time spent by the system in CP or supervisory mode.

2. User-states evaluation--this requires data on user-parameters such as the number of users in Q_1 and Q_2.

3. APL evaluation--this requires data on the user APL only without considering other users (see Fig. 3.2).

These three models are discussed in the next three sections. The methods, for statistical evaluation, are discussed in general terms in Sec. 4.5., and, the method used is described in more detail in Sec. 4.6.

Sec. 4.2. System Parameters

The control program accumulates certain statistics for the time-period during which the CP/67 system is operating on a particular day. The parameters for the evaluation of system performance were selected from these, and are listed below:

1. NUMUSERS: A half word (2 bytes) in memory is occupied by NUMUSERS. It is the number of users logged on the system at any instant. This, of course, represents the real external load (i.e. number of virtual machines) on the system.

2. CPTIME : A full word (4 bytes) in memory is occupied by CPTIME. CPTIME is the time spent by the system in CP mode; i.e. time spent by the system executing control program instructions in the supervisory state.

3. OVERHEAD: OVERHEAD occupies a full word in memory. OVERHEAD is the time spent (in DISPATCH) to find a new user to run after it has been determined that the current user is not runnable. This value, hence, is included in CPTIME as illustrated in Fig. 4.1.

4. WAITTIME: This occupies a full word in memory. WAITTIME is the time spent by the system when DISPATCH does not find a single runnable user. DISPATCH remains idle until an interrupt occurs (usually caused by change in some user's state).

5. PGREAD : PGREAD is the total number of pages read in since system initialization on a particular day.

6. PGSWAP : This is the total number of pages swapped out since system initialization on a particular day.

The parameters CPTIME, OVERHEAD, and WAITTIME indicate the fraction of the time the system spends in different modes. The time spent by the system in problem mode (this is not accumulated by the system) can be

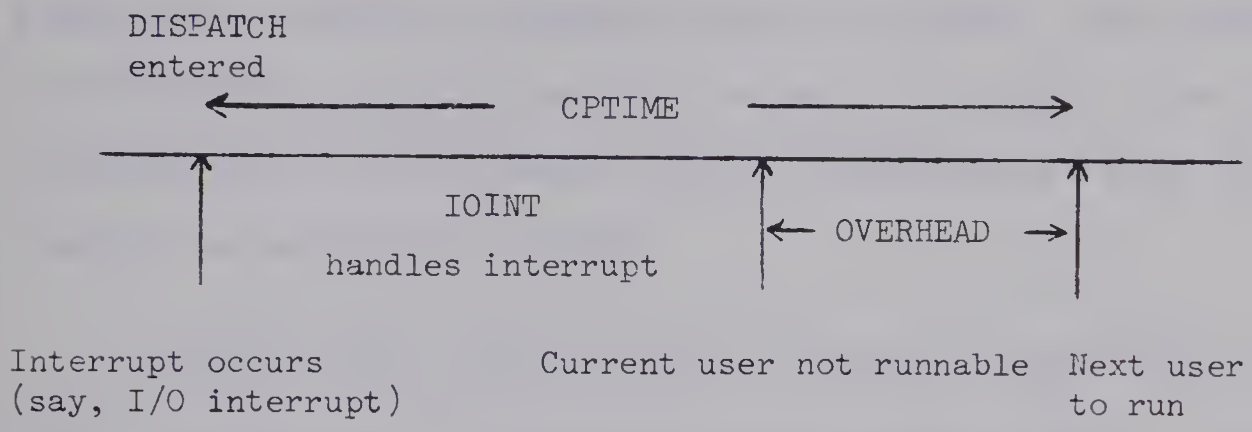

Fig. 4.1. Format of CPTIME

determined by the following relation:

$$\text{PROBTIME} = \text{TOTAL TIME} - (\text{CPTIME} + \text{WAITTIME}).$$

where PROBTIME is the time spent in problem mode and TOTALTIME is the time from system initialization to the time of observation.

The parameters OVERHEAD and CPTIME also indicate the efficiency of the main control routine DISPATCH. They are, in a way, measures of the scheduling policy. Too much time spent in the OVERHEAD part of CP mode would imply the ineffectiveness of the scheduling policy; although the blame should not fall on the scheduling algorithm entirely. Users can also degrade the performance of DISPATCH. The larger the number of "not-runnable" users, the more OVERHEAD and CPTIME accumulate. This

happens because DISPATCH, in trying to find the first "runnable" user, has to check all the "not-runnable" users in turn, thus taking more time. Similarly, users performing "I/O bound" jobs, or "non-compute" bound jobs would necessarily require more service from the CP supervisor.

The last two parameters occupy a full word each, in memory. Together, they indicate the paging load on the system. Since paging is done from disk, they also indicate the part of the load on the disk storage device, and the channel activity associated with the information transfer to and from this device.

The only other system parameter considered is the "real time of day" to determine the time observations are taken, and the interval between successive observations. This value is accumulated in "hours-minutes-seconds" only.

Sec. 4.3. Virtual Machine Parameters

The second evaluation model determines user-states through the following two entries in each user's UTABLE.

(i) TIMINQ; (ii) VMSTATUS.

TIMINQ is the accumulation of time spent by the user in Q_1 or Q_2. This is reset to zero, everytime he leaves Q_1 or Q_2. The following information can be obtained from the word TIMINQ:

If TIMINQ = 0, then the user is neither in Q_1 nor in Q_2.

If TIMINQ $\neq$ 0, then the user is either in Q_1 or in Q_2.

One important point to remember is that only the first 3 bytes of TIMINQ represent the value of the time spent in Q_1 or Q_2. The last byte of the

word is labelled NOQUANT and can contain any value from 0 to 127. NOQUANT is the count of the number of time slices used by the user, while in Q_2. In addition, the first bit of NOQUANT is labelled CONQBIT and is used to decide which queue the user is in (see Fig. 4.2).

If CONQBIT = 0, then the user is a member of, or eligible to enter Q_2.

If CONQBIT = 1, then the user is a member of, or eligible to enter Q_1.

The format of TIMINQ is illustrated in Fig. 4.2.

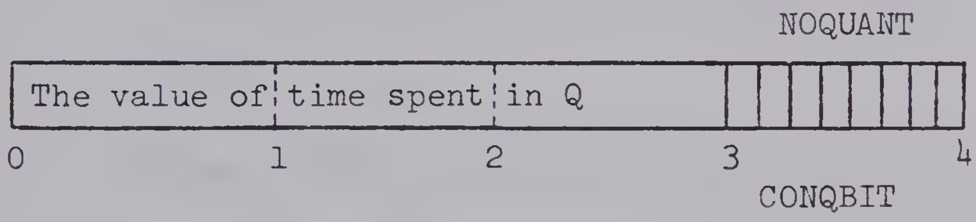

Fig. 4.2. TIMINQ

From now on, reference to TIMINQ would imply just the first three bytes of the word.

Combining TIMINQ and CONQBIT, the following deductions can be made:

If TIMINQ ≠ 0 and CONQBIT = 1, then the user is in Q_1.

If TIMINQ ≠ 0 and CONQBIT = 0, then the user is in Q_2.

If TIMINQ = 0 and CONQBIT = 1, then the user is eligible to enter Q_1 or the user is in console function wait (defined below).

Finally,

if TIMINQ = 0 and CONQBIT = 0, then the user is eligible to enter Q_2.

To determine the number of users eligible for Q_1 and to investigate user-states in more detail, the following parameter is necessary.

VMSTATUS indicates the state of the user's machine. There are eight possible states that can be tested but only the following four were selected (see Appendix A, UTABLE, for description of the other four states):

PGWT : Indicates whether or not the user is waiting for a page to be brought in from secondary storage (DISK) to main storage.

IOWT : Indicates whether or not the user is waiting for an I/O operation to be completed.

CFWT : Indicates whether or not the user is waiting to perform a console function. Obtaining this information helps in deciding whether the user is eligible for Q_1 or waiting to carry out a console function.

EXCFN : Indicates whether the user is actually executing a console function.

These four states are represented in the first byte of VMSTATUS by a "1" in first, second, third, or fourth bit, respectively.

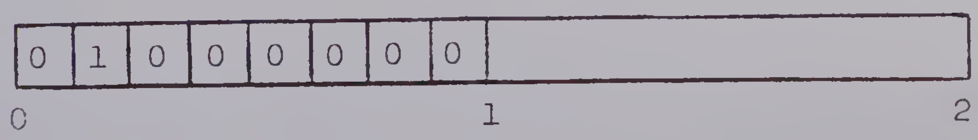

Fig. 4.3. VMSTATUS

VMSTATUS value shown in Fig. 4.3 implies that the corresponding virtual machine is in IOWT.

Hence TIMINQ and VMSTATUS determine users' states adequately (see Sec. 3.3).

For example, a user can have in his UTABLE,

1	0	0	0	0	0	0	0

VMSTATUS = the above,

TIMINQ $\neq$ 0, and CONQBIT = 1.

This would imply that the user is in Q_1 and waiting for a page to be brought in.

Sec. 4.4. APL-Users' Parameters

As mentioned before, at the time of observation APL represented at least 50% of the user population using CP (see Fig. 3.2). This classifies APL as a predominant user, and its evaluation deserves special treatment.

The APL-users' UTABLE is investigated in more detail to get a closer look at its performance. The following parameters are selected for this:

1. TIMINQ.

2. VMSTATUS.

3. TIMEUSED: This occupies a full word in the UTABLE; TIMEUSED is the

total time used since log on (problem state + CP overhead charged to APL). This gives the load APL puts on the CP/67 system as far as compute-time is concerned.

4. NUMPAGES and PRIORIT: These two together occupy a full word in the UTABLE. NUMPAGES is the number of pages APL has in core at a particular time instant. PRIORIT is important if APL ever enters Q_2. If it does, then this gives the priority of APL in Q_2.

5. VTOTTIME: This occupies a full word in the UTABLE. VTOTTIME is the total problem-state time, since log on, used by APL. Hence, (TIMEUSED-VTOTTIME) gives the CP overhead charged to APL.

All these parameters were selected for one basic reason, namely to indicate and classify the load APL puts on the CP/67 system, the first two have been described in Sec. 4.3. NUMPAGES and VTOTTIME indicate paging and CPU load, respectively, and TIMEUSED minus VTOTTIME gives the CP overhead charged to APL.

The number of users using APL is obtained from a fixed location in the memory (NUMDIAL). The variation of the state of the APL machine for the different number of users connected is important, because future prediction of the load APL can handle can be obtained through this.

Sec. 4.5. Possible Data Gathering Techniques

The selection of all the parameters for the evaluation models

are a direct result of the methods available to gather data on CP/67.

There are two approaches possible.

1. DISPATCH has a subroutine called, DEMON, which is used mainly for updating real time every second. There is an entry to DEMON every quarter second on the average. Hence after every four entries to DEMON, the time is updated. This routine, then, can be extended to gather statistics at each updating of real time.

The two main disadvantages of this method are the following.
First, it would require some "patching" or rewriting of the control program. A storage area would have to be kept free to accumulate different statistics. This would encroach on the free storage available to the control program. Moreover, "playing around" with the software of a system is always dangerous. The chances of bad linkages in inserting a "stat-gathering" program, are very high. In short, this would put unnecessary pressure on the control program.
Second, the gathering of user-data (2nd and 3rd part of the model) seems very difficult. UTABLE's are not in fixed location in the memory, but are "floating." They are linked together, but not necessarily in a contiguous memory block. The approach, through DEMON , then would make the system spent too much time in CP-mode trying to locate the required data.

One advantage of this method is that the time interval between successive readings can be made very small by activating DEMON more often

(say, everytime DISPATCH is entered). This again would result in a degradation of "user-response time" due to excessive time spent in CP-mode.

This method, then, was discarded for the above reasons and the following approach was used:

2. The other method available for such a statistical evaluation model uses a virtual machine (and hence is a "user" of CP/67) to look at the specified parameters periodically. The need for any "patching" of the control program becomes nonexistent. The load added to the system can be considered as just another user (virtual machine) using the system.

Through the use of the DIAG (Diagnose) instruction, all the necessary data can be obtained. The description of this method, and the program used to construct the evaluation model is given next.

Sec. 4.6. The Data Gathering Technique Used

The description of the DIAG instruction is given first in order to discuss the method more meaningfully.

DIAG is a privileged instruction (code '83'). To use it, a user requires the priority class A or C in the CP/67 system, where

class A - corresponds to the systems operator;

class B - corresponds to the systems administrator;

class C - corresponds to the subordinate systems operator; and

class D - corresponds to the system user.

This instruction is used for diagnostic purposes usually and hence, has the ability to look at specified memory locations. This ability is

used to gather statistics. The instruction is used in the following "360/Assembler" sequence of instructions:

```
LA    1, LIST
LA    2, NOLIST
LA    3, STLIST
DIAG  1, 3
```

The address of the list of memory locations to look at (LIST) is loaded in Register 1. The number of entries in this list (NOLIST) is loaded into Register 2 and the address of the storage area (STLIST) where the values from the specified memory locations, are to be stored, is loaded in Register 3. Then DIAG instruction obtains these values and stores them in the specified storage area.

The virtual machine with the identification name CPSTATS was used for this model. The priority class C was assigned to it.

A program CPSNOOP (listed in Appendix B) was written in 360/ Assembler and used some of the CMS macros available. The instruction DIAG was used throughout the program to gather statistics. Algorithm 4.1 and the flowchart in Fig. 4.4 describe CPSNOOP in detail indicating the method used to perform the required tasks for the first two models.

Algorithm: 4.1.

Step 1 Advance the buffer pointer 32 bytes to allow space for the information to be brought in. If the buffer area (800 bytes) will not handle the additional 32 bytes, write the information already

in the buffer and set the pointer to 32 before going to step 2.

Step 2 Use DIAG instruction to pick up the system parameters listed in Sec. 4.2 and store the values in the buffer area. Go to step 3.

Step 3 Use RUNUSER (UTABLE link) to load the current user's (in this case stat-gatherer--CPSTATS) UTABLE and again use DIAG instruction to pick up the pointer to NEXTUSER in the UTABLE chain. Go to step 4.

Step 4 Load next user's UTABLE, pick up user parameters as listed in Sec. 4.3 and also the "link" to the next user. Move pointer in the buffer to 8 bytes further and test the "buffer-full" condition as in step 1. If buffer can handle 8 bytes more, store information in buffer, otherwise, write the information in the buffer on disk and continue after resetting the pointer in the buffer to 8 bytes. Go to step 5.

Step 5 Test if the next user is the RUNUSER (CPSTATS); if it is, then go to step 6; if it is not then go to step 4.

Step 6 Perform some console function (fixed) and then go to step 1.

The step that ends this program is not included because it depends on the size of the sample required.

The purpose of performing the console function in step 6 is to keep the stat-gathering virtual machine (CPSTATS) in the high priority queue, Q_1, and also to control the time interval between successive readings. The virtual machine waits for this console function (in this

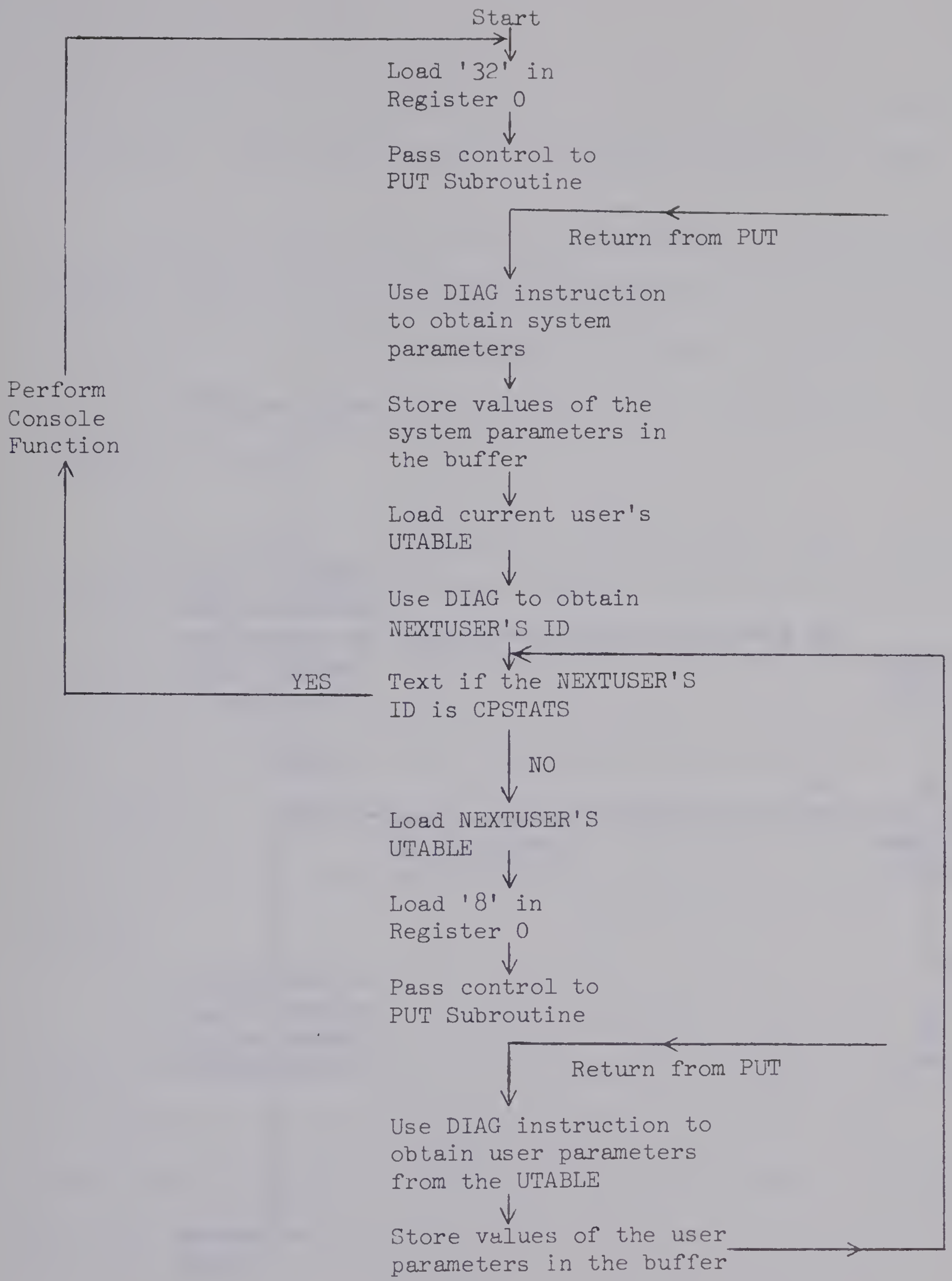

Fig. 4.4. Flowchart of CPSNOOP
Part I of II

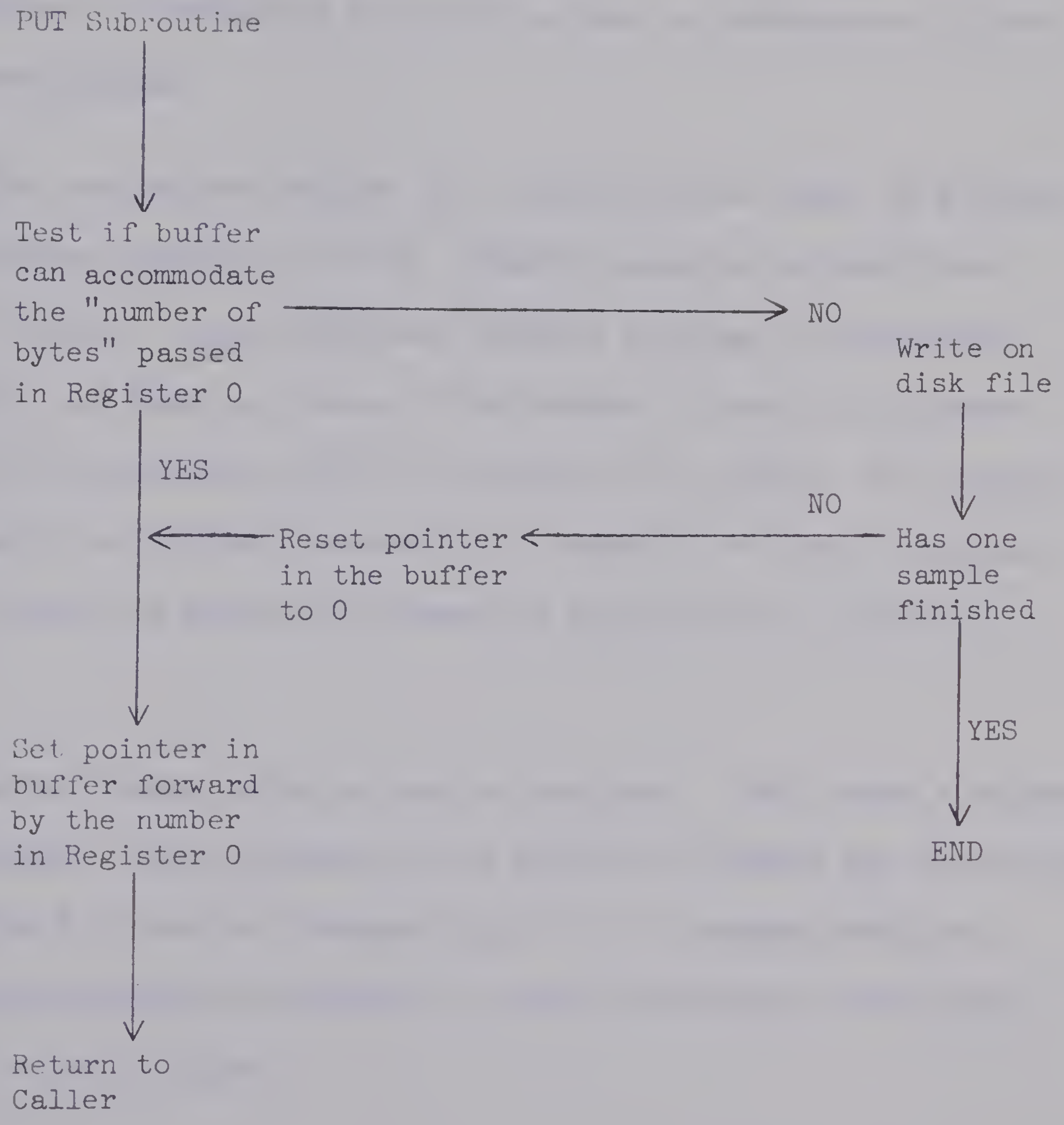

Fig. 4.4. Flowchart of CPSNOOP
Part II of II

case, 3 sequences of "space-backspace"), and then picks up system parameters again. This "wait" results in readings taken every second.

A buffer area of 800 bytes is also created to store the information gathered, sequentially. When this area gets full, the information is written out on disk before proceeding any further. A pointer is moved in the buffer as information is stored so that no information is lost through overlapping.

After running the program for a specified time (say, 20 minutes), another program ANALYSE is called. ANALYSE converts the data from binary to decimal, tests TIMINQ and VMSTATUS readings to decide the user-states, and makes up counts of the number of users in different states. This information, then, is punched out on cards. For each set of readings (i.e. for each interval of 1 second), one card is punched. Table 4.1 gives the format of information punched out on cards by ANALYSE.

The third model which is used to evaluate APL, needs a separate program APLDATA. This program is very similar to CPSNOOP and is described in Algorithm 4.2. and the flowchart Fig. 4.5. A program ANAPL was written (quite similar to ANALYSE) to convert these data into decimal and punch it out on cards.

Algorithm: 4.2.

Step 1 Advance the buffer pointer 12 bytes to allow space for the information to be brought in. If the buffer area (800 bytes)

TABLE 4.1

FORMAT OF INFORMATION PUNCHED OUT ON CARDS BY ANALYSE

Card Columns	Variable
1- 6	Time of day in seconds
7- 9	Number of users
10-16	CPTIME in seconds
17-23	WAITTIME in seconds
24-30	OVERHEAD in seconds
31-37	Number of pages read in
38-44	Number of pages swapped out
45-47	Number of users in page wait
48-50	Number of users in I/O wait
51-53	Number of users in Console function wait
54-56	Number of users executing Console function
57-59	Number of users in Q_2
60-62	Number of users in Q_1
63-65	Number of users eligible to enter Q_2
66-68	Number of users eligible to enter Q_1

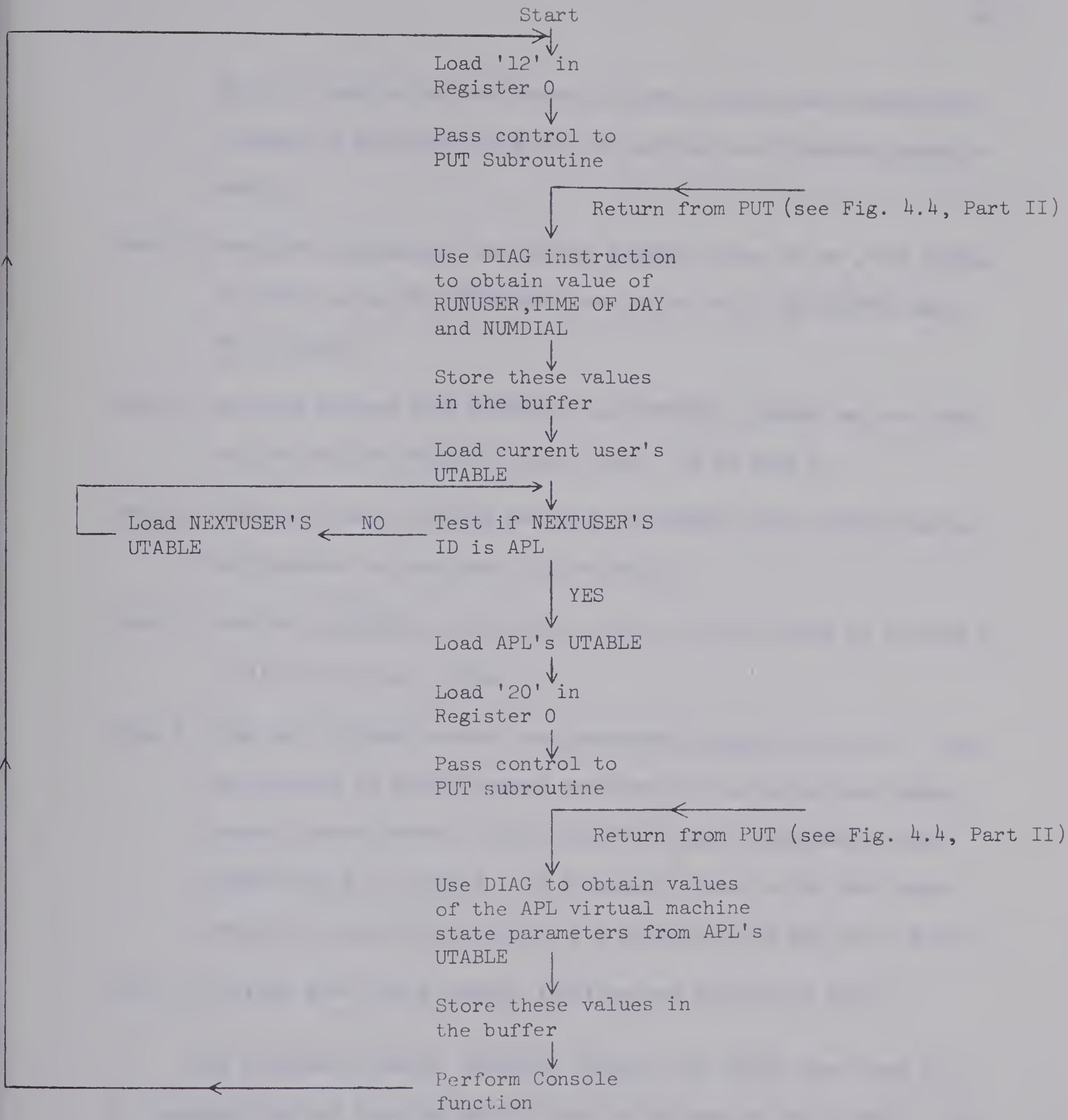

Fig. 4.5. Flowchart of APLDATA

will not handle the additional 12 bytes, write the information already in the buffer and set the pointer to 12 before going to step 2.

Step 2 Use DIAG instruction to pick up RUNUSER, time of day, and number of users using APL (NUMDIAL), and store it in the buffer area. Go to step 3.

Step 3 Get the address from RUNUSER, load CPSTAT's UTABLE and use DIAG to pick up the pointer to next user. Go to step 4.

Step 4 Load next user's UTABLE and pick up USERID (user identification) and pointer to next user. Go to step 5.

Step 5 Test this USERID to see if it is APL; if it is then go to step 6; if it is not, go to step 4.

Step 6 From APL's UTABLE select the parameters listed in Sec. 4.4. Set the pointer 20 bytes further and test if the buffer can handle these 20 more bytes. If it can, store this information from UTABLE and go to step 7. If it cannot, then, write the information in the buffer on disk, set pointer to 20 and go to step 7.

Step 7 Perform some fixed console function and then go to step 1.

The programs CPSNOOP, ANALYSE, APLDATA, and ANAPL were used for a one-month period from the end of June to the end of July, 1969. The CP/67 system ran from 3 p.m. to 7 p.m. every day. An average user population of about 8 to 10 users existed during most of these four hours. The high numbers of users were logged-on between 3.30 p.m. and 4.30 p.m. usually. A low value of number of users was found between 6 p.m. and

7 p.m. with the lowest value of 3 users obtained only after 6.30 p.m. The next chapter describes the physical restrictions on the models and the data obtained using these models. These data are analyzed in detail, and conclusions are drawn.

CHAPTER V

ANALYSIS OF RESULTS

The analysis of the data collected for a period of one month is given in this chapter. The samples consisted of 1200 observations, which were taken at one-second intervals for twenty minutes. This sample size proved to be convenient as far as restricting the mass of data and keeping the number of users nearly constant in each sample. The amount of data had a physical restriction in that only six cylinders of disk storage space were available. This space was used to store the data before they were punched onto cards. During the month of data-gathering, the CP/67 system ran for four hours every day. On the average, eight users (virtual machines) were logged-on for two to three hours. To obtain observations for other values of the number of users, "twenty-minute" samples were found to be effective. A low number of users usually existed only for the last twenty minutes of the time the CP/67 system was "on." The change in the number of users of the APL virtual machine was very rapid. Hence, to obtain a sample with a constant number of APL users, the time interval for sample observations was set to ten minutes. The analysis of the APL observations is given in Sec. 5.6.

Sec. 5.1 describes the transformations applied to each of the variables used to estimate the system and user parameters. The

randomness of the samples is discussed in Sec. 5.2, while the actual analysis of the data is described in Sections 5.3, 5.4, and 5.5.

Sec. 5.1. Transformations of Variables

Considering the first and second model, the values of the system variables (see Sec. 4.2) - Time of day; NUMUSERS, CPTIME, WAITTIME, OVERHEAD, PGREAD, PGSWAP - and the user variables (see Sec. 4.3) - TIMINGQ and VMSTATUS - were obtained at one-second intervals. The logical tests on TIMINQ and VMSTATUS, described in Sec. 4.3, yielded the following eight variables:

1. Number of users in PGWT;
2. Number of users in IOWT;
3. Number of users in CFWT;
4. Number of users executing console function (EXCFN);
5. Number of users in Q_1;
6. Number of users in Q_2;
7. Number of users eligible for Q_1 (EQ1);
8. Number of users eligible for Q_2 (EQ2).

These logical tests are part of the program ANALYSE, described in Sec. 4.6.

The following transformations of variables were performed before the data on punched cards were written onto tape. Time of day was checked first to see if the readings were taken at one-second intervals. It was found that after every n seconds, where n was

different for different samples, a reading was skipped. Instead of one-second, a two-second time interval between successive readings was observed. Whenever the program CPSNOOP had to write the data from the buffer onto disk storage, "data-gathering" stopped, and, thus, one reading was skipped. The lengths of the one-or-two-second intervals were used in the rest of the transformations described below.

Instead of using the actual values of CPTIME, WAITTIME, and OVERHEAD, the percentages of the time the system spent in the respective modes were calculated. Each of these were obtained by subtracting the respective value at the time instant i from the value at the next time instant i+1, dividing this by the length of the time interval (i,i+1), and then multiplying by 100. The values of PGREAD and PGSWAP were also calculated for the time interval (i,i+1) and then divided by the length of the time interval to obtain the paging activity per second. PROBTIME percentages were calculated by subtracting the (CPTIME + WAITTIME) percentages from 100. Thus, the values of the transformed variable PROBTIME are the percentages of the time the system spends in the problem mode. Throughout the rest of this chapter, the "transformed" variables will carry the name of the "actual" variables.

The user variables were transformed as follows: Each variable was divided by NUMUSERS and then multiplied by 100 to obtain the percentage of the total number of users in a particular state. During the month when the "data-gathering" was carried out, the average number of users was about eight.

Percentage values were considered for the following reasons: Suppose that the number of users logged-on to the system in a particular twenty-minute session was found to be 5 and the number of users in Q_1 also was found to be 5; and in the next sample, 10 users were logged-on, and again 5 users were found in Q_1. Then to deduce that the number of users in Q_1 is constant and equal to 5 would be nonsensical. A percentage value would give a better measure for comparisons. The actual value should be used if a maximum value of the number of users in a particular state is to be predicted. In this study the actual values were not considered since the number of users was too low.

The above described percentage values of the fifteen system and user variables were written out on tape for each time instant. They were written separately for each sample so that any analysis program could be directed to any particular sample, and only to that sample, if needed.

Sec. 5.2. Test for Randomness

The first step taken to analyze the data was to test the sample observations for random behaviour. Testing for randomness is difficult because there is no overall definition of "random." Tests for certain criteria of nonrandom behaviour can be considered. Then the rejection of the hypothesis of nonrandomness supports the alternate hypothesis of random behaviour of the series of observations considered. Some of the characteristics of nonrandomness of a series of observations are:

1. The presence of extreme variations,
2. the presence of trends,

3. the presence of periodic fluctuations,
4. the presence of discontinuities.

Assuming that the observations are a random sample from a distribution with a continuous cumulative density function, Wald and Wolfowitz (30) (see also, Bennet and Franklin (2), page 687) studied the distribution of the random variable $R'_h = \sum_{i=1}^{n} x_i \, x_{i+h}$. For R'_1 they showed that

$$\text{mean } (R'_1) = E(R'_1) = \frac{1}{n-1}\,(S_1^2 - S_2),$$

and

$$\text{Var } (R'_1) = \sigma^2_{R'_1} = \frac{S_2^2 - S_4}{n-1} + \frac{S_1^2 - 4S_1^2 S_2 + 4S_1 S_3 + S_2^2 - 2S_4}{(n-1)\,(n-2)} - \frac{(S_1^2 - S_2)^2}{(n-1)^2}$$

where $S_K = \sum_{i=1}^{n} x_i^K$

and n is the number of observations. The distribution of R'_1 is shown to approach normality as n increases. Hence, the significance of observed R'_1-values can be tested for sufficiently large n by considering T,

$$T = \frac{R'_1 - E(R'_1)}{\sigma_{R'_1}},$$

as a standardized normal random variable.

To avoid repetitions, only the continuous variable CPTIME was tested for samples 1 to 14 and the values of T were tabulated in Table 5.1. The upper 95% critical point of a standardized random variable is 1.645. Hence the hypothesis of random behaviour of the observations is accepted for all the samples.

TABLE 5.1

STANDARDIZED VALUES OF RANDOM VARIABLE R_1'
AS APPLIED TO CPTIME

Sample	T	Sample	T
1	0.166	8	0.072
2	0.203	9	0.795
3	0.113	10	0.233
4	0.278	11	0.092
5	0.162	12	0.231
6	0.064	13	0.125
7	0.284	14	0.081

Sec. 5.3. Tabulations of Sample Statistics

Two types of tabulations of sample statistics (mean, standard deviation, maximum value and minimum value) were carried out. The first set of 15 tables (see Tables I in Appendix C) are the sample statistics for each of the 15 variables, for the 16 samples separately. The second set of tables (see Tables II in Appendix C) gives the mean values of the variables for different numbers of users regardless of the sample they come from. The only restriction applied in the second case was that the mean values were calculated for only those numbers of users for which there were more than 100 observations in a sample.

Frequency histograms were also obtained for each variable and all samples to get a picture of the shape of the distributions. These

histogram-plots are given in Figures H1 to H14 in Appendix D. NUMUSERS is the only variable which is not plotted since the variation associated with this variable is small.

In the next section, a systematic discussion of the implications of the above mentioned sample statistics for each variable is given.

Sec. 5.4. Discussion of Sample Statistics

1. NUMUSERS - (see Table I(a) Appendix C).

The minimum of all the observed values of NUMUSERS was 3 and the maximum was 15. Hence there existed a user (virtual machine) population varying from 3 to 15 users during the time of data-gathering. On the average there were 8 users logged-on to the system during that time. The maximum and minimum values indicate the range of the number of users in each sample. For example in the sample 15, observations are taken for only 3 and 4 users whereas in sample 8, the number of users observed were 8 to 9.

2. CPTIME - (see Table I(b) Appendix C)

An overall average of about 18% was obtained. Thus on the average, the system spent 18% of its time in the supervisor or CP-mode. The high values of the standard deviations suggest that any conclusions drawn from the average value would have to include the standard error of the mean. A minimum value of 1% and a maximum value of 100% can also be read off this table. The Fig. H1 shows the observed behaviour of CPTIME. Using the sample mean and variance, both a Poisson and a Normal distribution were fitted to these data, but were rejected since

the Kolmogorov-Smirnov (see Feller (12)) test indicated a poor fit. The third and fourth moments were also calculated. A negative value of kurtosis indicated the presence of a smaller number of large deviations than would be expected if the distribution were normal.

3. WAITTIME - (see Table I(c) Appendix C)

The overall observed average was 27%. This implies that 27% of the total time the central processing unit was idle. This is a very high value, implying inefficient use of the machine by the users. WAITTIME is user controlled since a high value is caused by users not utilizing the CPU facility to the full extent. For example, in sample 9, a value of 85.6% is obtained for a user population varying from 6 to 15 users. Although one would expect a high value of WAITTIME for a low user population, this is not always so as Tables I(a) and I(c) indicate. It may be more meaningful to report two average values for this variable. There is either a low value as in samples 1, 3, 5, 7, 8, 10, 12, 14, 15, 16, which on the average is 7.3%; or a very high value as for samples 2, 4, 6, 9, 11, 13, which on the average is 59.9%. The low average WAITTIME's were plotted as a histogram, Fig. H2(a), in Appendix D, and are all concentrated in the interval (0,5)%. Fig. H2(b) indicates the behaviour of the high average WAITTIME's.

4. OVERHEAD - (see Table I(d) Appendix C)

The overall average was 6% overhead in the observed time intervals. The maximum value of OVERHEAD among all the samples was 30% which occurred in sample 11. Looking at Table I(a), the average number of users logged-on when sample 11 was obtained, was 12 to 14, which is one

of the highest among all the samples. This does not necessarily imply any relation between NUMUSERS and OVERHEAD. The correlation between these two variables was calculated and found to be very small for all the samples. Correlation studies are discussed in greater detail in Sec. 5.5 and Sec. 5.6. A chi-square distribution with mean 6 was fitted to the OVERHEAD values, and the Kolmogorov-Smirnov test was used to test the fit. A poor fit was indicated.

A new variable to consider at this point is (CPTIME - OVERHEAD), which can be called the "actual CP-mode" time. Referring back to Figure 4.1, this is the time the system spends in supervisor mode handling CP requests (i.e. handling interrupts). An overall average of 12% was obtained. This implies that one eighth of the total time is spent handling interrupts.

The low average OVERHEAD value of 6% implies high efficiency of the scheduling algorithm, since this is the time the actual scheduling takes place (see Figure 4.1). This average OVERHEAD percentage was obtained for an average of 8.3 users, which can be considered as a low load on the system. The constant nature of the average value over all samples suggests that this variable is "user-independent" and an increase in the number of users may not affect it to any great extent.

5. PROBTIME - (see Table I(e) Appendix C)

This variable was described indirectly when the two variables CPTIME and WAITTIME were discussed, but the importance of the variable warrants special attention. The time the system spends in the problem

mode can be considered one of the most important parameters as far as calculating system efficiency or utilization is concerned. Studying the Table I(e), 10 out of the 16 samples gave a value of around 70% or more. Interestingly enough the maximum value never went beyond 97.5%. Thus, the maximum efficiency obtained from the system was 97.5%. As for the variables CPTIME and WAITTIME, the sample standard deviations of PROBTIME are large. The histogram in Fig. H13 in Appendix D, indicates the shape of the observed frequency distribution. A normal distribution was fitted to these data but was rejected since again, the Kolmogorov-Smirnov test indicated a poor fit.

6. PGREAD and PGSWAP - (see Tables I(f) and I(g) Appendix C)

These values are discussed together because of the great similarities between them. An overall average of 1.1 pages per second read in, and 1 page per second swapped out, was obtained. In sample 14, a maximum value of 49 pages per second read in and 38 pages per second swapped out, was found. This is an extremely large transfer of information to and from disk storage. A transfer of 49 pages (196K bytes, where 1K = 1024 bytes) should put a very heavy load on the selector channel used. Since a selector channel transfers 312,000 bytes per second, it can be seen that the transfer of 196K bytes would take 0.65 seconds. An important measure obtained from these two tables is the total number of pages read in plus the total number of pages swapped out, which is called the "total paging load" here. The maximum paging load, 9034 pages, in sample 16 implies that a maximum (during the time of observation) of 36136K bytes of information are transferred through the selector channel in twenty

minutes. Again, using the data transmission rate for the selector channel, it would take two minutes to do this. Looking at sample 15 (3 users) and sample 16 (12 to 14 users) it can be seen that a very low value of PGREAD and PGSWAP was obtained for sample 15 while an extremely high total value was obtained for sample 16. A high correlation between NUMUSERS and paging load is indicated. Figures H4 (PGREAD) and H5 (PGSWAP) both show a concentration of values in the interval (0,2) pages per second and otherwise are nearly alike.

7. PGWT - (see Table I(h) Appendix C)

An overall average of only 0.8% of the total number of users were found in PGWT at the instant of observation. The maximum and minimum values were 62.5% (sample 1) and 0%, respectively. Fig. H6 indicates a concentration of values of the variable in the interval (0,2)% and otherwise very few extreme values were observed.

8. IOWT - (see Table I(i) Appendix C)

An overall average of 2.58% was obtained for this variable. The maximum and minimum values were 100% (sample 15) and 0%, respectively. Again, large values of the standard deviations were found. Fig. H7 shows that over 80% of values are concentrated in the interval (0,2)%. Some values were also found in the intervals (8,10)% and (18,20)%.

9. CFWT - (see Table I(j) Appendix C)

On the average, 5.7% of the total number of users were found to be in "console function wait." The maximum and minimum values were 71.43% (sample 4) and 0%, respectively. Fig. H8 confirms that the

variation of this variable is very large indicating the extreme fluctuations.

10. EXCFN - (Executing Console Function)

The value of this parameter was always found to be zero. The reason is that a user can only be executing console function if the system is in supervisory state. At the time of observation, the virtual machine CPSTATS is running and hence the system is in the problem state.

11. EQ1 and Q1 - (see Tables I(k) and I(l) Appendix C)

The percentage of users eligible for Q_1 was about 76%, which is very much higher than the percentage of users (4%) actually in Q_1 at the instant of observation. The one-second time interval between observations appears to be too large. In order to obtain a more realistic distribution of the number of users in Q_1 (i.e. the queue length distribution) values of Q_1 must be obtained at intervals very much smaller than one second. The "dispatch algorithm" only allows a user's request to remain in Q_1 until it has been given its first CPU time-slice of less than 0.05 seconds in one burst. The success of such a scheduling scheme is based on giving good response to users whose requests are satisfied by one short burst of CPU time. In a console-oriented interactive system a large number of user requests are of this type. The difference between the average percentages of users eligible for Q_1 (EQ1) and that of users in console function wait (CFWT) was about 70%. This implies that 70% of the users can be classed as "low execution" users. This value, it must be remembered, also includes users who are logged-on but have not requested service. This is substantiated by the high values of WAITTIME in Table I(c). Over 85% of users were found to be eligible for Q_1 in

all the samples with high values of WAITTIME. The maximum value, 100%, occurred in sample 7 for which three to seven users were logged-on. Otherwise nearly all samples showed that 50% or less of the users were in Q_1. The frequencies of the EQ1 and Q1 values were plotted in Fig. H9 and H10, respectively. Fig. H10 indicates that nearly all Q1 observations are in the interval (0,2)%.

12. EQ2 and Q2 - (see Tables I(m) and I(n))

On the average about 2% of the users were eligible for Q_2 and about 17.2% were actually in Q2. The relatively higher value of Q2 can be attributed to the nature of user requests in Q2. They represent requests for more than 0.05 seconds of CPU time usually. The smaller value of EQ2 is a result of the very small amount of time spent in that state. From Tables I(e), I(m) and I(n), respectively, it can be seen that high PROBTIME values occur in the samples which have large percentages of users in Q_2 and eligible to enter Q_2. Similarly, the samples with low PROBTIME values correspond to low Q2 and EQ2 values. Also, high PROBTIME values imply low values of WAITTIME. Thus, the system seems to be able to use up "idle time" by running a Q_2 user. Fig. H11 (Q2) indicates a concentration of values in the intervals (16,20)% and (36,40)%. These account for about 70% of the values. Fig. H12 (EQ2) shows a concentration of values in the interval (0,2)%. Some values are also found in the interval (8,10)%.

In the rest of this section, a discussion of the series of tables referred to as Table II is given. Instead of discussing each table separately, the relation of some of the system variables with the number

of users is considered. The following sample correlation coefficients were found:

TABLE 5.2

CORRELATIONS BETWEEN THE NUMBER OF USERS AND CPTIME, WAITTIME, OVERHEAD, AND PGLOAD

Variable	Correlation
CPTIME	0.932
WAITTIME	0.002
OVERHEAD	0.004
PGLOAD	0.951

In Fig. 5.1 the mean values of PGLOAD (PGREAD + PGSWAP) are plotted against the number of users, see Table II(a) to II(g). Plotting $\log_e$ (PGLOAD) against the number of users, see Fig. 5.2, nearly produces a straight line, which suggests an exponential relationship between these two variables. To test this hypothesis a linear regression analysis was done on $\log_e$ (PGLOAD) (say, y) against NUMUSERS (say, x). The following results were found obtained.

The regression equation was found to be

$$y = .58x - 4.9,$$

where the standard error of the slope is 0.076, and the correlation of x and y is 0.96. Transforming y to e^y, the PGLOAD values may be predicted by the equation

$$\text{PGLOAD} = 0.0075\, e^{.58x},$$

TABLE 5.3

NUMUSERS AGAINST PGLOAD AND CPTIME

NUMUSERS	PGLOAD	$\log_e$(PGLOAD)	CPTIME	$\log_e$(CPTIME)
3	0.015	-4.20	12.5	2.52
4	0.15	-1.90	13.7	2.62
7	0.56	-0.58	15.0	2.71
8	1.50	0.41	21.0	3.05
10	2.91	1.07	19.1	2.95
11	4.77	1.56	21.5	3.07
13	8.50	2.14	25.5	3.24

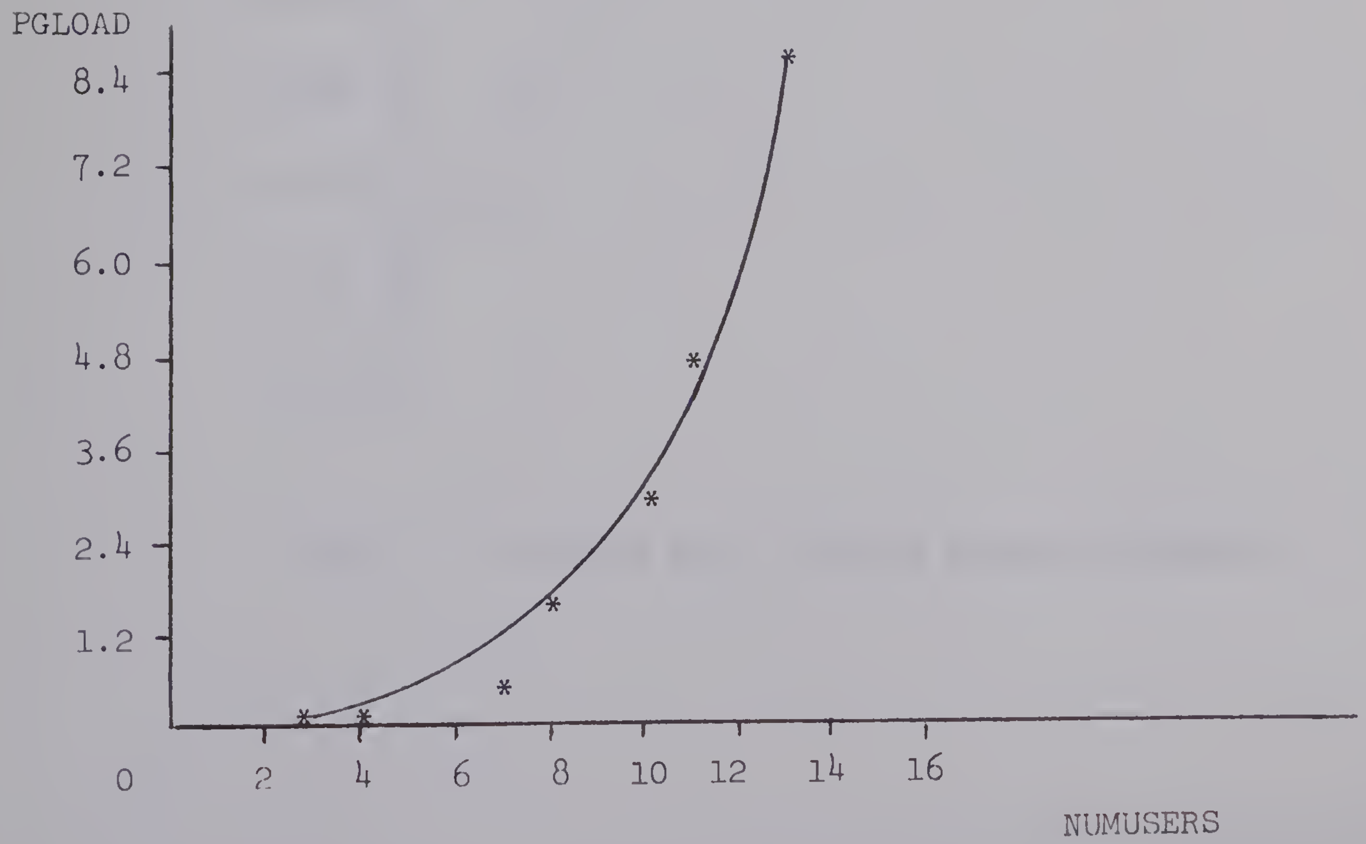

Fig. 5.1. Plot of PGLOAD Against NUMUSERS

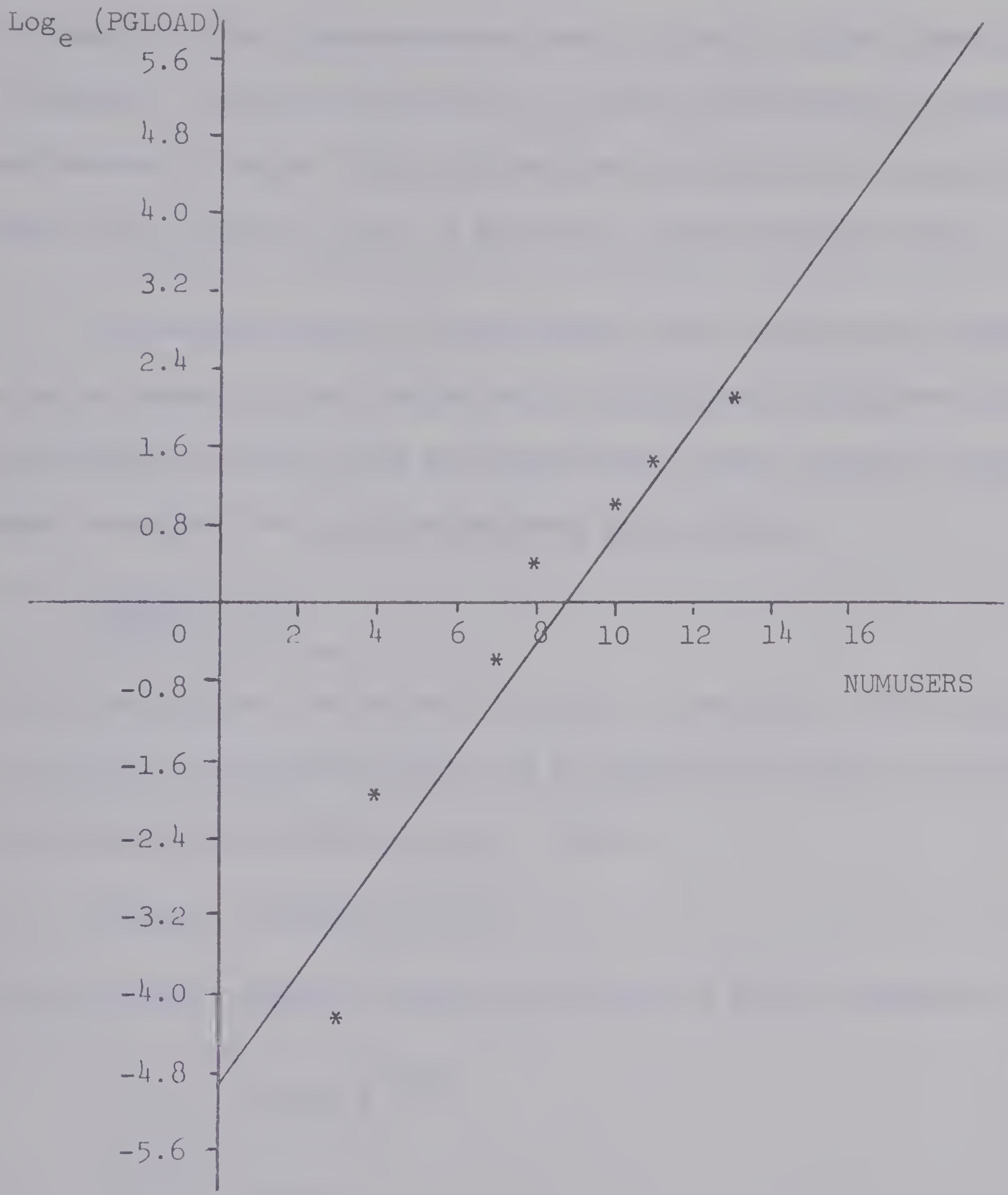

Fig. 5.2. Plot of Log$_e$ (PGLOAD) Against NUMUSERS

where x is the number of users. The observed F-value in the analysis of variance of the regression was found to be 58.3 with 1 and 5 degrees of freedom. The critical value $F_{1,5}(\cdot 01) = 16.26$ which is less than the observed F-value. This implies that the predicted values of PGLOAD using this relation produce a good fit to the observed values.

The maximum number of pages that can be transferred through the selector channel in one second can be calculated as follows: The channel can transfer 312,000 bytes per second and 1 page consists of 4096 bytes; hence the number of pages transferred per second is

$$\frac{312000}{4096} = 76.2.$$

Thus assuming that the selector channel is available 100% of the time for paging, this maximum value can be used to calculate the corresponding value of the number of users. Since,

$$\text{PGLOAD} = 0.0075\ e^{0.58x},$$

for the maximum number of pages per second it can be seen that

$$76.2 = 0.0075\ e^{0.58x},$$

$$e^{0.58x} = 10140,$$

$$0.58x = \log_e (10140)$$

$$= 9.2252$$

and $$x = 15.91.$$

Hence, for about 16 users, the channel would be used for paging 100% of the time. In actual fact the channel is not available 100% of the time

for this purpose. A great amount of time is usually spent in seeking the required data from disk and during this time the channel is inactive. In actual fact then, a queue for the selector channel service should build up for a smaller number of users.

In Table 5.3 the mean values of CPTIME are given for different values of the number of users, (also see Tables II(a) to II(g)). The plot of $\log_e$(CPTIME) against the number of users, see Figure 5.3, gives a straight line, which again indicates an exponential relationship between these two variables. This hypothesis was tested by performing a linear regression analysis on $\log_e$(CPTIME) (say, y) and NUMUSERS (say, x). The following results were obtained.

The regression equation was found to be

$$y = 0.07\,x + 2.34,$$

where the standard error of the slope is 0.01, and the correlation of x and y is 0.94. Transforming y to e^y, the CPTIME values can be predicted by the equation

$$\text{CPTIME} = 10\,e^{0.07x},$$

where x is the number of users. The F-value in the analysis of variances of the regression was found to be 39.02 with 1 and 5 degrees of freedom. This value, again, is greater than the critical value $F_{1,5}(0.01)$. Hence CPTIME values may be predicted to give a good fit to the observed values, using the above equation.

For example, for x = 20,

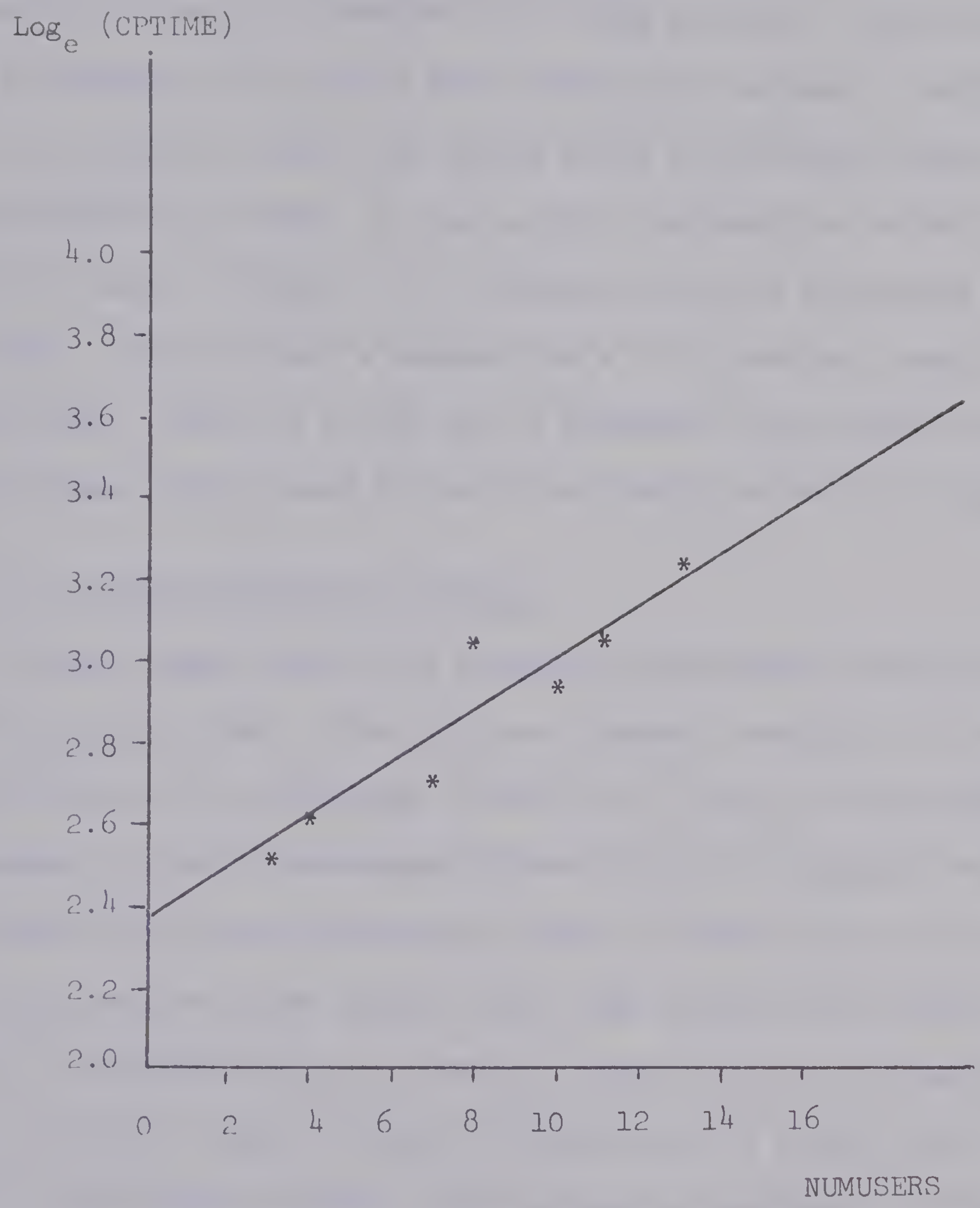

Fig. 5.3. Plot of Log$_e$ (CPTIME) Against NUMUSERS

$$\text{CPTIME} = 10\ e^{1.4}$$

$$= 40\%.$$

Calculating the value of NUMUSERS for CPTIME at 100%, a value of 33 users is obtained. This value seems unrealistic because it implies that with 33 users or more, the system would be performing supervisory functions 100% of the time. It may be that the behaviour of the curve beyond the range in Figure 5.4 is different from the calculated exponential shape. It could have a maximum for $x < 33$ users and remain steady at this value. There is no way such a judgement can be made with the available data. This could be one of the future projects to study.

Sec. 5.5. Further Regression Analysis

At this stage some of the expected relationships between the variables are discussed. The first and foremost relation to be studied is the effect of the percentage of users in Q_1 and Q_2 on the PROBTIME percentages. A large percentage of users in Q_1 (or eligible for Q_1) would reduce the system efficiency, unless of course, there are enough users in Q_2 who would use up the "idle" time caused by too much console activity. PROBTIME should be inversely proportional to the number of users in the wait states, whereas, CPTIME should be proportional to this number. The PAGELOAD variable could perhaps be expressed in terms of the users in page wait. Stepwise regression analyses were carried out to study the relationships discussed above. An independent variable was allowed to enter the regression if it produced a reduction in the error sum of squares of more than 1%. A sample output of these analyses

is given in Appendix E.

1. PROBTIME as the dependent variable

Regression analysis was performed separately for all the samples. The percentage of users eligible for Q_2 and in Q_2 entered the regression in all the samples. Interestingly enough, the "wait-state" variables never entered the regression. This again implies that a runnable user must have been available to the system most of the time. At the 1% level, the variable Q1, also, did not enter the regression, signifying the small contribution it makes to system performance as far as high utilization of the CPU is concerned. A high value (over 0.7 on the average) of multiple correlation was obtained for all the samples. Different values of regression coefficients were obtained for each sample. The following relation, from sample 7, indicates the nature of the relationship:

$$\text{PROBTIME} = 4.4\ (Q2) + 4(EQ2) - 0.06\ [(Q2)^2 + (EQ2)^2] + 10.4.$$

Thus, a number of Q2 users are necessary among all the users to run the system efficiently.

2. "CP mode time" as the dependent variable, where CP mode time = (CPTIME - OVERHEAD)

A relation between CP mode time and the three main supervisory functions of CP was obtained by regression. These three functions are: (i) performing I/O, i.e. the percentage of users in I/O wait; (ii) paging, i.e. PGLOAD; and (iii) dealing with console requests, i.e. percentage of users in CF wait. This relation should be treated cautiously because a low multiple correlation (0.57 on the average) was obtained.

3. PGREAD as the dependent variable

A high correlation (> 0.94) between PGREAD and PGSWAP was calculated. The following relationship was obtained:

$$PGREAD = c + b\ (PGSWAP),$$

where c was found to be 0.04, on the average, and b was a little greater than 1. An example of this relationship, from sample 13, is

$$PGREAD = 0.02 + 1.09\ (PGSWAP).$$

Thus, the value of PGREAD was normally found to be greater than that of PGSWAP. This is due to pages being swapped only if a request for a new page cannot be satisfied for lack of main storage space to fit the requested page (see Sec. 3.4). Hence, if main storage is not completely utilized, the value for PGSWAP will be less than that of PGREAD. It appears that the main storage is usually fully utilized and that these two variables are equal.

4. PGLOAD as the dependent variable

The following variables entered the regression: (i) PGWT, (ii) Q1, and (iii) CP mode time.

The entry of variable Q1 implies that users in Q_1 make most of the paging requests. Since Q2 did not enter the regression, users in Q_2 do not require a significant amount of paging. The entry of PGWT and CP mode time in the regression was as expected. A low value (0.57 on the average) of the multiple correlation coefficient was obtained.

A similar result to (4) was obtained for IOWT as the dependent

variable. Again, users in Q_1 make more I/O requests than users in Q2.

Sec. 5.6. APL Results

To make the discussion of the APL results more meaningful, reference is made constantly to the series of tables called Table III in Appendix C and to the histograms in Figures H14 to H17 in Appendix D. Results from APLDATA were written onto tape with the percentage values of TIMEUSED and VTOTTIME (see Sec. 4.4) calculated. The observed values of NUMPAGES, PRIORIT, and NUMDIAL (number of users dialled-on to the APL virtual machine) were written out on tape without having transformed these values. The state of the APL virtual machine at every second was written on tape in the following way: Considering the eight user-states; (i) PGWT, (ii) IOWT, (iii) CFWT, (iv) EXCFN, (v) Q2, (vi) Q1, (vii) EQ2, and (viii) EQ1, an eight digit number, with a '1' in the position corresponding to the state of the virtual machine at that instant and '0' in every other position, was written. Thus, if APL was in IOWT and in Q1, the eight bit number would be 01000100. Thus, at every instant of observation each of the eight APL-state variables either had the value 1 or 0.

The sample statistics, mean, standard deviation, maximum, and minimum values of NUMDIAL, TIMEUSED, VTOTTIME, and NUMPAGES, were calculated for the six samples (see Tables III(a) to III(d)). The value of PRIORIT was found to be equal to zero in all the samples. Significance of this is discussed later in this section. The values of the APL-state variables were added separately for each sample; divided by the total

number of observations in the respective sample; and, then multiplied by 100 to obtain the total percentage of time the APL virtual machine was in the different states (see Table III(e)).

Instead of discussing each variable separately a joint description is given here. From Table III(a), a range of 6 to 17 users dialled to the APL virtual machine was found. An overall average of 13 users existed at the time of observation. Utilizing the VTOTTIME percentages, the average percentage of CPU time used by the APL machine was 30.1% (see Table III(c)). The supervisor on the average was used 4.7% of the time, which was obtained from the (TIMEUSED - VTOTTIME) percentages. This implies that the APL machine was performing I/O or was idle 65.2% of the total time. The APL machine, hence, can be described as an "I/O bound" user. From Table III(d), the APL machine has on the average 54 pages in core. The overall maximum value of NUMPAGES was found to be 91. Thus, a maximum of 91 × 4 = 364K bytes of main storage was used, which classifies APL as a "heavy" core user. Even the overall minimum value of 29 pages, in sample 5, occupies 116K bytes in the main storage. Fig. H17 shows a typical plot of NUMPAGES with some concentration of values in the intervals (24,28) pages and (28,32) pages. In Fig. H15 for the variable TIMEUSED, the main concentration of values was found in the interval (0,5)%. A similar concentration in the interval (0,5)% was found for the variable VTOTTIME (see Fig. H16). A chi-square distribution was tried to fit the observed values of (TIMEUSED - VTOTTIME) (see Fig. H14) but the fit turned out to be poor.

On the average both Q_1 and Q_2 contained APL virtual machine

requests 33% of the time (see Table III(e)). Whereas the APL user was eligible for Q_1 27% of the time, it was eligible for Q_2 only 7% of the time. The scheduling algorithm was designed to keep the APL requests in Q_1, but this does not seem to be the case at the time. Since the observed value of PRIORIT (APL's priority in Q_2) (see Sec. 4.4) is always zero, an APL request does not seem to remain in Q_2 for long. This value of PRIORIT implies that the APL virtual machine does not use 0.05 seconds of CPU time in one burst while in Q_2. If it did, a nonzero value of PRIORIT would be observed. Hence, the apparent flaw of the scheduling algorithm is not serious. Some of the users of the APL machine may have "compute-bound" programs which are forced into Q_2 by using 0.05 second in one burst, while in Q_1. From Table III(e) it can be seen that on the average, the percentages of time spent in PGWT and IOWT are 0.75, 3.3. Whereas the values of CFWT and EXCFN could not be found by the method user. The low percentages may be attributed to the length of the time interval between observations. Another reason for these small values may be that the 360/APL system schedules the execution of its "ready" users while another of its users is in one of the above wait states.

From the above results, the APL virtual machine can be characterized as follows: The APL requests are console oriented (27% of the time in the EQ1 state), I/O bound (only 34.8% of the time are the CPU and the CP supervisor used) and core storage demanding (on the average 54 pages are in core, which is equivalent to 212K bytes of main storage). The final conclusion, an assessment of the achievements of these models and guidelines for further research, are discussed in the next chapter.

CHAPTER VI

CONCLUSION AND SUGGESTIONS FOR FURTHER STUDIES

The main aim of this study was to obtain some kind of evaluation measures of the time-sharing system, CP/67. One of the measurements obtained was the average CPU utilization (PROBTIME) value of about 55% whereas the maximum CPU utilization was found to be about 97.5%. A synopsis of the system performance as observed during the specified data-gathering period is given below:

TABLE 6.1

SUMMARY OF SYSTEM PARAMETERS

Percentage of time in	Overall Average	10 of 16 samples with high PROBTIME	Samples with minimum WAITTIME	Highest number of Users	APL Machine
CP mode	12.0	12.8	13.6	18.1	4.7
Overhead	6.0	6.9	7.3	7.1	-
Probtime	55.0	73.0	78.6	70.2	30.1
Waittime	27.0	7.3	0.5	4.6	65.2
Number of Users	8.3	8.1	10.2	12.5	12.9

From Table 6.1, it is unrealistic to expect an average problem time to be greater than 75%. To discuss the total overhead of the supervisor both the interrupts charged to the respective virtual machines (CP mode) and the actual recorded overhead must be added. The above measures only describe CPU time utilization without considering the core utilization. An overall average of 2 pages were read in and swapped out every second. Just the fact that the pages were swapped every second implies that the main storage was used to capacity. In Sec. 5.4 it was found that PGLOAD increases exponentially with an

increasing number of users. In reporting system performance it must be realized that these measurements are user dependent.

The "user states" parameters describe the general behavior of users or try to characterize the user population. The console oriented nature of the users is very evident through the usually high values of EQ1 (see Sec. 5.4). Considering the regression analysis in Sec. 5.5 it was found that the users in Q_2 do not reach wait-states as often as the users in Q_1 and hence, may be classed as "long execution" users. An overall average of 17% of the users were of this type. Another user characteristic appears to be that a fairly high percentage of them sit at their terminals "thinking" and force the system into a wait-state. The scheduling algorithm should not be blamed for this. The high average values of WAITTIME appear to be entirely due to the users.

From Table 6.1 it is clear that APL applies a very heavy load on the CP system. Problem mode time of 30.1% is approximately half of the total CP problem mode, and since the range of average values is from 10% to 54% there are some cases when it is well over half. Moreover, APL was a heavy user of core storage using an average of 212K bytes with a range from 116K to 364K bytes.

One of the most important achievements of this study is the method used to obtain the data. This method is unique in that it does not involve making any internal changes to the system. The load imposed by the data-gathering virtual machine was similar to the load by any other user and therefore did not bias the results significantly. Thus, the method seems to be an ideal one to use for further research.

Let it be stressed that a group of users classed as "the average user" is considered not to exist. The only solution that seems possible is to smooth the series of observations by attaching "weights" according to the user-population they are obtained from. An extensive study of the "type" of users should be carried out to classify them in different groups and associate "weights" according to the demands they make on the system. For example, I/O bound and compute-bound users can be classified in such a way. It might even be necessary to divide compute-bound users into subclasses according to their CPU time requirements. This would allow a selection of user classes which could be used as bench marks in future system evaluation. Using the overall average problem time of 55% with 7 CP users and average APL problem time of 30.1% with 13 users from Table 6.1, it is conjectured that, on the average, 25% of the problem time was spent servicing the 7 CP users. If it is assumed that the problem time cannot exceed 75% it would appear that 13 CP users and the APL machine (with 13 users) would produce the maximum utilization (highest problem time). If there are only CP users then it is conjectured that the maximum utilization would be obtained with 21 users; if there are only APL users it is conjectured that the maximum would occur with 31 users. For a large system like 360/67 to support only 21 CP users or 31 users of a single language, it is clearly very expensive.

Finally to sum up the achievements of this project it is suggested that an important new method of obtaining data is discovered. Various performance measures describing the system are obtained, which are useful in evaluating the time-sharing system CP/67.

REFERENCES

1. Adiri, I., and Avi-Itzhak, B., "A Time-sharing Queue With a Finite Number of Customers," *Oper. Res. Stat. and Econ. Mimeo. Series No. 10*, Dept. of Indust. and Mgt. Eng., Technion, Israel Inst. Technol., Haifa, Israel (February) 1968.

2. Bennett, C.A., and Franklin, N. L., *Statistical Analysis in Chemistry and the Chemical Industry*. John Wiley & Sons Inc., New York, Third Printing (May) 1963.

3. Cantrell, H., "Talk presented at Workshop on Models for Time-shared Processing in Symposium on Computer and Communication, Their System Interaction," Sponsored by the IEEE Communications Technology Group and the Computer Group, Santa Monica, Calif. (January) 1967. (Unpublished but reported in Ref. 11.)

4. Chang, W., "A Queueing Model for a Simple Case of Time-sharing," *IBM Syst. J.* 5, 115-125 (1966).

5. _______, "Queues With Feedback for Time-sharing Computer System Analysis," *Oper. Res.* 16, 613-627 (1968).

6. Coffman, E. G., and Krishnamoorthi, B., "Preliminary Analysis of Time-shared Computer Operation," Doc. SP-1719, Sys. Development Corp., Santa Monica, Calif. (August) 1964.

7. _______, "Stochastic Models of Multiple and Time-shared Computer Operations," Unpublished Ph.D. dissertation, Dept. of Eng., U.C.L.A., Los Angeles, Calif. (June) 1966.

8. _______, "An Analysis of Computer Operations Under Running-time Priority Disciplines," *Proc. ACM Symp. on Interactive Sys. for Exptl. and Applied Maths.* Academic Press, New York, 1968.

9. _______, and Kleinrock, L., "Feedback Queueing Models for Time-shared Systems," *J. ACM* 15, 4, 549-576 (October 1968).

10. Corbató, F. J., *et al.*, "An Experimental Time-sharing System," *Proc. AFIPS 1962 Spring Joint Comput. Conf.*, Vol. 21, pp. 335-344, Spartan Books, New York.

11. Estrin, G., and Kleinrock, L., "Measures, Models and Measurements for Time-shared Computer Utilities," *Proc. of 22nd Natl. Conf., Aug. 1967*, pp. 85-96, Thompson Book Co., Washington, D.C.

12. Feller, W., "On the Kolmogorov-Smirnov Limit Theorems for Empirical Distributions," *Ann. of Math. Stat.* 19, 177-189.

13. Fife, D., and Rosenberg, R., "On Queueing in a Memory Shared Computer," Report from Cooley Electronic Laboratory, Univ. of Michigan, Ann Arbor, (1966).

14. Fine, G. H., and McIsaac, P. V., "Simulation of a Time-sharing System," SDC Rept. SP-1909 (December) 1964.

15. Greenberger, M., "The Priority Problem and Computer Time-sharing," *Man. Sci.* 12, 888-906 (1966).

16. Kleinrock, L., "Analysis of a Time-shared Processor," *Naval Res. Log. Quart.* 11, 59-73 (1964).

17. _______, "Time-shared Systems: A Theoretical Treatment," *J. ACM* 14, 2, 242-261 (April 1967).

18. Krishnamoorthi, B., and Wood, R. C., "Time-shared Computer Operations With Both Interarrival and Service Times Exponential," *J. ACM* 13, 3, 317-338 (July 1966).

19. _______, "The Stationary Behaviour of a Time-sharing System Under Poisson Assumptions," Doc. SP-2090, Sys. Development Corp., Santa Monica, Calif. (September) 1965.

20. Lavita, P. P., "Digital Simulation of a Time-shared System," Unpublished Master's thesis in Mathematics, New York Univ. (February) 1966.

21. Nielson, N. R., "The Analysis of General Purpose Computer Time-sharing Systems," Unpublished Ph.D. dissertation, Dept. of Computing Sci., Stanford Univ., Stanford, Calif., (1966).

22. Patel, N. R., "A Mathematical Analysis of Computer Time-sharing Systems," Unpublished Master's thesis, Dept. of Elec. Eng., M.I.T., Cambridge, Mass. (May) 1964.

23. Scherr, A. L., *An Analysis of Time-shared Computer Systems.* M.I.T. Press, Cambridge, Mass., 1967.

24. Schrage, L. E., "The Queue M/G/1 With Feedback to Lower Priority Queues," *Man. Sci.* 13, 466-474 (1967).

25. _______, and Miller, L. W., "The Queue M/G/1 With the Shortest Remaining Processing Time Discipline," Rand Corp. Rept. P 3263 (November) 1965.

26. _______, "Some Queueing Models for a Time-shared Facility," Unpublished Ph.D. dissertation, Dept. of Indust. Eng., Cornell Univ. (February) 1966.

27. Shemer, J. E., "Some Mathematical Considerations of Time-sharing Scheduling Algorithms," *J. ACM* 14, 2, 262-272 (April 1967).

28. Sutherland, G. G., "Talk presented at Workshop on Models for Time-shared Processing in Symposium on Computer and Communication, Their System Interaction," Sponsored by the IEEE Communications Technology Group and the Computer Group, Santa Monica, Calif. (January) 1967. (Unpublished but reported in Ref. 11.)

29. Totchek, R. A., "An Empirical Investigation into the Behaviour of the SDC Time-sharing System," SDC Rept. Sp-2191 (August) 1965.

30. Wald, A., and Wolfowitz, J., "An Exact Test for Randomness in the Nonparametric Case Based on Serial Correlation," *Ann. Math. Stat.* 14, 378-388 (1943).

APPENDICES

APPENDIX A

PRIMARY USEP CONTROL TABLE

```
UTABLE     DSECT            USER PRIMARY CONTROL TABLE
VGPRS      DS     16F       GENERAL PURPOSE REGISTERS
VFPRS      DS      4D       FLOATING-POINT REGISTERS
VPSW       DS      1D       USER'S CURRENT PSW
SEGTABLE   DS      1F       POINTER TO SEGMENT TABLE
VMACHSIZ   DS      1F       SIZE OF VIRTUAL MACHINE
VCHSTART   DS      1F       START OF VIRTUAL CHANNEL LIST
VCHCOUNT   DS      1H       COUNT OF VIRTUAL CHANNELS FOR THIS USER
PENDING    DS      1H       PENDING INFORMATION
VMACHNO    DS      1H       VIRTUAL MACHINE NUMBER
VMSTATUS   DS      1H       VIRTUAL MACHINE STATUS
TIMEUSED   DS      1F       TIME USED SINCE LOGIN
NEXTUSER   DS      1F       POINTER TO NEXT USER CONTROL TABLE
VTIMER     DS      1F       VIRTUAL INTERVAL TIMER
USERID     DS      1D       EXTERNAL USER IDENTIFICATION
CPRQUEST   DS      1F       ADDRESS OF STACKED CP ROUTINE REQUEST
USYSTAB    DS      1F   POINTER TO TABLE FOR SHARED SYSTEM PAGES
VMXSTART   DS      1F       START OF MULTIPLEXOR LIST
VMXPOINT   DS      1F       POINTER TO INTERRUPTING DEVICE
VMFLAGS    DS      0D       MISCELLANEOUS FLAGS
VIOLOCK    DS      1C       VIRTUAL I/O TABLE LOCK
CPRQLOCK   DS      1C       CPRQUEST USAGE LOCK
CIOLOCK    DS      1C       CONSOLE I-O QUEUE LOCK
PRCLASS    DS      1C       PRIVILEGE CLASS AND PRIORITY OF USER
VSCHQUE    DS      1F
CIOREQ     DS      1F       POINTER TO CONSOLE I/O REQUEST STACK
NCIOREQ    DS      1H       NUMBER OF PENDING CONSOLE I/O REQUESTS
CPRQCNT    DS      1H       NUMBER OF CONTROL PROGRAM EXECUTION REQ.
VMXCOUNT   DS      1H       COUNT OF MPX SUBCHANNEL DEVICES
SEGTBDSP   DS      1H       DISPLACEMENT OF SEGTABLE FROM START
```

PRIMARY USER CONTROL TABLE (Continued)

```
PAGEORG   DS    1F                    USER'S PAGING ORIGIN.
TIMEON    DS    6C
TIMERMOD  DS    1C                    TIMER MODE SWITCH
PAGWCNT   DS    1C                    PAGEWAIT COUNT = NUM. OF PAGES IN TRANSIT
ACCTNG    DS    1D
TIMINQ    DS     F                   FOR DISPATCH--TO LIMIT MULTIPROGRAMMING
NOQUANT   EQU    TIMINQ+3                 NO OF QUANTA USED WHILE IN Q
CONQBIT   EQU    TIMINQ+3
CONSOLQ   EQU    X'80'
NUMPAGES  DS     H                        NUM OF PAGES USER HAS IN CORE
PRIORIT   DS     H                        PRIORITY TO REENTER Q
VTOTTIME  DS     F
WORKSET   DS     C

          DS    3C                    UNUSED
UVIOCNT   DS     H                    VIRTUAL SIO COUNT
          DS    3H                    UNUSED

          BITS DEFINED IN VMSTATUS ---

          BYTE 0 ---

PAGEWAIT  EQU    X'80'                USER IS WAITING FOR A PAGE
IOWAIT    EQU    X'40'                USER HAS PENDING I/O OPERATION
CFWAIT    EQU    X'20'                USER HAS PENDING C.F. OPERATION
EXCFN     EQU    X'10'                VM EXECUTING CONSOL FUNCTIONS
UARPQ     EQU    X'08'                USER ALLOWED TO REQUEST PAGES
VIRCOMSW  EQU    X'04'
INLOGOFF  EQU    X'02'
INLOGON   EQU    X'01'

          BYTE 1 ---

                          USED VIA 'TS' INSTRUCTION TO INDICATE MACHINE RUNNING
```

PRIMARY USER CONTROL TABLE (Continued)

```
          BITS DEFINED IN PRCLASS ---
          COMMAND PRIVILEGE CLASSES (IN HIGH ORDER HALF OF BYTE)

SYSCTLOP  EQU     X'80'                    SYSTEM CONTROL OPERATOR        CLASS A
SYSADMIN  EQU     X'40'                    SYSTEM ADMINISTRATOR           CLASS B
SUBSYSOP  EQU     X'20'                    SUBORDINATE SYSTEM OPERATORS   CLASS C
SYSUSER   EQU     X'10'                    SYSTEM USER                    CLASS D

          USER PRIORITY 0-9 IN LOW ORDER HALF OF BYTE

                  BITS DEFINED IN 'TIMERMOD'

DISCNBIT  EQU     X'80'                    USER CONSOL DISCONNECTED
PRIDISP   EQU     X'40'                    PRIORITY DISPATCH FOR THIS TIME
MSGBIT    EQU     X'20'                    IGNORE MESSAGES BIT
WNGBIT    EQU    X'10'                  IGNORE WARNINGS BIT
MULTCH    EQU     X'08'                    MULTIPLE VIRTUAL CH WITH SAME ADDRESS
RUNCP     EQU     X'04'                    VM RUNNING WITH CF READ ACTIVE
  BITS 6-7 USED BY USEROFF
  BIT 6 ON : PRIORITY LOGOFF
  BIT 7 ON : NO LOGOFF MESSAGE
UTABLESZ  EQU     (*-UTABLE)/8             SIZE OF UTABLE (DOUBLE WORDS)

          PRINT ON,NOGEN
```

APPENDIX B

LISTING OF THE FOUR PROGRAMS USED IN THE EVALUATION MODEL IS GIVEN HERE

I) CPSNOOP

```
         MACRO
&L       DIAG  &R1,&R3,&CODE=X'0004'
&L       DC    X'83',AL1(16*&R1+&R3),&CODE
         MEND
CPSNOOP  START
         LR    12,15
         USING CPSNOOP,12
         ST    14,SAVE14
         CMSREG
         LA    0,4                        TEST FOR 4 MORE BYTES IN BUFFER
         BAL   14,PUT
         MVC   0(4,R3),=C'STRT'           INDICATE START NEW FILE
AGAIN    EQU   *
         LA    0,32                       TEST FOR 32 MORE BYTES IN BUFFER
         BAL   14,PUT
         LA    1,RUNLIST
         LA    2,8
         DIAG  1,3                        OBTAIN VALUES OF SYSTEM PARAMETERS
         LM    10,11,0(3)
         LR    9,10
         USING UTABLE,9
REPEAT   EQU   *
         LA    0,TIMINQ
         LA    1,VMACHNO                  SET UP USER PARAMETERS
         LA    2,NEXTUSER
         STM   0,2,UTABLIST
         LA    0,8                        TEST FOR 8 MORE BYTES IN BUFFER
         BAL   14,PUT
         LA    1,UTABLIST
```

(CONTINUED)

```
          LA    2,3
          DIAG  1,3                                       OBTAIN VALUES OF USER PARAMETERS
          BCT   11,STILMORE
          LA    15,4
          B     EXIT
STILMORE  L     9,8(R3)                                   IS NEXT USER CPSTATS
          CR    9,10
          BNE   REPEAT                                    NO - BRANCH TO REPEAT
          LA    0,4                                       YES - TEST FOR 4 MORE BYTES IN BUFFER
          BAL   14,PUT
          MVC   0(4,R3),=C'EOIN'                          INSERT END OF INFORMATION TAG
          LA    1,TYPEPL
          SVC   X'CA'                                     PERFORM CONSOLE FUNCTION
          DC    AL4(*+4)
          B     AROUND                                    SEQUENCE OF 3 'SPACE-BACKSPACE'
TYPEPL    DS    OD
          DC    CL8'TYPE'
          DC    AL1(1)
          DC    AL3(*+7)
          DC    C'K'
          DC    AL3(6)
          DC    3XL2'4016'
          DS    OH
AROUND    LA    1,WAITPL
          SVC   X'CA'
          DC    AL4(*+4)
          B     AGAIN                                     BRANCH TO AGAIN
WAITPL    DS    OD
          DC    CL8'WAIT'
          DC    CL4'CON1'
          DC    F'O'
          DS    1F
FCB       MADDPL NAME=STATS,TYPE=DATA,BUFF=BUFFER,SIZE=800
PUT       L     2,SAVENEXT
          LR    3,2                                       POINTER TO SPACE IN BUFFER IN REGISTER 2
          AR    2,0
          ST    2,SAVENEXT                                IS BUFFER FULL
```

(CONTINUED)

```
         C     2,ENDBUFR
         BCR   4,14                       YES - INSERT END OF BUFFER BLOCK
         MVC   0(4,R3),=C'EOBB'
         WRBUF FCB,ERROR=PUTERR           WRITE ONTO DISK FILE
         L     7,COUNT                    IS ONE SAMPLE FINISHED
         LA    7,1(7)                     NO - CONTINUE
         ST    7,COUNT
         C     7,LIMIT                    YES - QUIT
         BC    10,QUIT
         LA    2,BUFFER                   RESET POINTER IN BUFFER
         LR    3,2
         AR    2,0
         ST    2,SAVENEXT
         BR    14
PUTERR   LA    15,8
         B     EXIT
QUIT     SR    15,15
EXIT     L     14,SAVE14
         BR    14
SAVE14   DS    F
RUNLIST  DC    A(RUNUSER)
         DC    A(NUMUSERS)
         DC    A(CPTIME)
         DC    A(WAITTIME)
         DC    A(OVERHEAD)
         DC    A(PGREAD)
         DC    A(PGSWAP)
         DC    A(HOURS)
UTABLIST DS    OD
         DS    A
         DS    A
         DS    A
SAVENEXT DC    A(BUFFER)
ENDBUFR  DC    A(BUFEND)
BUFFER   DS    CL800
BUFEND   EQU   *
         DS    CL4
```

(CONTINUED)

```
NUMUSERS EQU   X'997C'
CPTIME   EQU   X'9914'
COUNT    DC    F'1'
LIMIT    DC    F'201'
ONE      DC    F'1'
TEMPSAVE EQU   X'160'
RUNUSER  EQU   TEMPSAVE+64
HOURS    EQU   RUNUSER+8
PGREAD   EQU   HOURS+36
PGSWAP   EQU   PGREAD+4
WAITTIME EQU   PGSWAP+12
OVERHEAD EQU   WAITTIME+4
         COPY  UTABLE
         END
```

II) ANALYSE

```
ANALYSE  START
         LR    12,15
         USING ANALYSE,12
         ST    14,SAVE14
         CMSREG
         STATE SFCB,NOFILE
         SETUP FCB
         RDBUF FCB,ERROR=INITERR                   READ FIRST 800 BYTES IN BUFFER
         LA    3,BUFFER
NOW      EQU   *
         LA    0,4                                 SET POINTER IN BUFFER TO 4
         BAL   14,GET
         UNPK  CARD(7),28(4,R3)                    TIME OF DAY IN COL. 1 TO 6
         L     10,4(R3)
         SRA   10,16
         CVD   10,TEMP1
         OI    TEMP1+7,X'OF'
         UNPK  CARD+6(3),TEMP1+6(2)                NUMUSERS IN COL. 7 TO 9
```

(CONTINUED)

```
         L      11,8(R3)
         SRA    11,8
         M      10,CONST2
         D      10,DIVISOR
         CVD    11,TEMP1
         OI     TEMP1+7,X'OF'
         UNPK   CARD+9(7),TEMP1+4(4)              CPTIME IN COL. 10 TO 16
         L      11,12(R3)
         SRA    11,8
         M      10,CONST2
         D      10,DIVISOR
         CVD    11,TEMP1
         OI     TEMP1+7,X'OF'
         UNPK   CARD+16(7),TEMP1+4(4)             WAITTIME IN COL. 17 TO 23
         L      11,16(R3)
         SRA    11,8
         M      10,CONST2
         D      10,DIVISOR
         CVD    11,TEMP1
         OI     TEMP1+7,X'OF'
         UNPK   CARD+23(7),TEMP1+4(4)             OVERHEAD IN COL. 24 TO 30
         L      11,20(R3)
         CVD    11,TEMP1
         OI     TEMP1+7,X'OF'
         UNPK   CARD+30(7),TEMP1+4(4)             PGRD IN COL. 31 TO 37
         L      11,24(R3)
         CVD    11,TEMP1
         OI     TEMP1+7,X'OF'
         UNPK   CARD+37(7),TEMP1+4(4)             PGSWP IN COL. 38 TO 44
         LA     0,40                              SET POINTER IN BUFFER 40 BYTES FORWARD
         BAL    14,GET
AGAIN    EQU    *
         L      10,0(R3)                          TEST IF TIMINQ=0
         SRA    10,8
         BC     8,CHECK                           YES - BRANCH TO CHECK
         TM     3(R3),X'80'                       NO - TEST IF CONQBIT=0
         BC     8,Q2                              YES - BRANCH TO Q2
```

(CONTINUED)

```
         L     10,NOINQ1
         LA    10,1(10)                NO - ADD ONE TO NUMBER IN Q1
         ST    10,NOINQ1
         B     MOREIN                  BRANCH TO MOREIN
Q2       L     10,NOINQ2
         LA    10,1(10)                ADD ONE TO NUMBER IN Q2
         ST    10,NOINQ2
         B     MOREIN                  BRANCH TO MOREIN
CHECK    TM    3(R3),X'80'             TEST IF CONQBIT=0
         BC    8,QUE2                  YES - BRANCH TO QUE 2
         L     10,ELIGQ1
         LA    10,1(10)                NO - ADD ONE TO NUMBER ELIGIBLE FOR Q1
         ST    10,ELIGQ1
         B     MOREIN                  BRANCH TO MOREIN
QUE2     L     10,ELIGQ2
         LA    10,1(10)                ADD ONE TO NUMBER ELIGIBLE FOR Q2
         ST    10,ELIGQ2
MOREIN   TM    6(R3),X'80'             TEST VIRTUAL MACHINE IN
         BZ    *+16                    DIFFERENT USER STATES
         L     10,PW
         LA    10,1(10)                PGWT COUNT
         ST    10,PW
         TM    6(R3),X'40'
         BZ    *+16
         L     10,IOW                  IOWY COUNT
         LA    10,1(10)
         ST    10,IOW
         TM    6(R3),X'20'
         BZ    *+16
         L     10,CFW                  CFWT COUNT
         LA    10,1(10)
         ST    10,CFW
         TM    6(R3),X'04'
         BZ    *+16
         L     10,CFN                  EXCFN COUNT
         LA    10,1(10)
         ST    10,CFN
```

(CONTINUED)

```
MORE     LA     0,8                          SET POINTER IN BUFFER 8 BYTES FORWARD
         BAL    14,GET
         CLC    0(4,R3),=C'EOIN'             TEST FOR END OF INFORMATION TAG
         BNE    AGAIN                        NO - BRANCH TO AGAIN
         L      11,PW                        YES - CONTINUE
         CVD    11,TEMP1
         OI     TEMP1+7,X'OF'
         UNPK   CARD+44(3),TEMP1+6(2)        PGWT IN COL. 45 TO 47
         L      11,IOW
         CVD    11,TEMP1
         OI     TEMP1+7,X'OF'
         UNPK   CARD+47(3),TEMP1+6(2)        IOWT IN COL. 48 TO 50
         L      11,CFW
         CVD    11,TEMP1
         OI     TEMP1+7,X'OF'
         UNPK   CARD+50(3),TEMP1+6(2)        CFWT IN COL. 51 TO 53
         L      11,CFN
         CVD    11,TEMP1
         OI     TEMP1+7,X'OF'
         UNPK   CARD+53(3),TEMP1+6(2)        EXCFN IN COL. 54 TO 56
         L      11,NOINQ2
         CVD    11,TEMP1
         OI     TEMP1+7,X'OF'                Q2 IN COL. 57 TO 59
         UNPK   CARD+56(3),TEMP1+6(2)
         L      11,NOINQ1
         CVD    11,TEMP1
         OI     TEMP1+7,X'OF'
         UNPK   CARD+59(3),TEMP1+6(2)        Q1 IN COL. 60 TO 62
         L      11,ELIGQ2
         CVD    11,TEMP1
         OI     TEMP1+7,X'OF'
         UNPK   CARD+62(3),TEMP1+6(2)        EQ2 IN COL. 63 TO 65
         L      11,ELIGQ1
         CVD    11,TEMP1
         OI     TEMP1+7,X'OF'
         UNPK   CARD+65(3),TEMP1+6(2)        EQ1 IN COL.66 TO 68
         LA     1,PLIST
```

(CONTINUED)

```
         SVC   X'CA'                                    PUNCH A CARD
         LTR   15,15
         BNZ   PCHERR
         LA    10,NOINQ2
         XC    0(32,10),0(10)
         B     NOW                                      BRANCH TO NOW
PCHERR   LA    15,4
         B     EXIT
EXIT     L     14,SAVE14
         BR    14
FCB      MADDPI NAME=STATS,TYPE=DATA,BUFF=BUFFER,SIZE=800
GET      AR    3,0
         CLC   0(4,R3),=C'EOEB'
         BCR   7,14
         RDBUF FCB,ERROR=GETERR
         LA    3,BUFFER
         BR    14
GETERR   CKEOF REALERR
         LA    15,10
         B     EXIT
REALERR  LA    15,8
         B     EXIT
NOFILE   LA    15,1
         B     EXIT
INITERR  LA    15,12
         B     EXIT
SFCB     DS    OD
         DC    CL8'STATE'
         DC    CL8'STATS'
         DC    CL8'DATA'
         DC    CL2'P1'
         DC    H'0'
         DC    F'0'
PLIST    DS    OD
         DC    CL8'CARDPH'
         DC    A(CARD)
SAVE14   DS    F
```

(CONTINUED)

```
BUFFER   DS    CL800
TEMP     DS    D
TEMP1    DS    D
CARD     DS    CL80
DIVISOR  DC    F'300'
CONST2   DC    F'100'
NOINQ2   DC    F'0'
NOINQ1   DC    F'0'
ELIGQ2   DC    F'0'
ELIGQ1   DC    F'0'
PW       DC    F'0'
IOW      DC    F'0'
CFW      DC    F'0'
CFN      DC    F'0'
         END
```

III) APLDATA

```
         MACRO
&L       DIAG  &R1,&R3,&CODE=X'0004'
&L       DC    X'83',AL1(16*&R1+&R3),&CODE
         MEND
APLDATA  START
         LR    12,15
         USING APLDATA,12
         ST    14,SAVE14
         CMSREG
         LA    0,4                            TEST FOR 4 MORE BYTES IN THE BUFFER
         BAL   14,POINT
         MVC   0(4,R3),=C'STRT'               INSERT START FILE TAG
REPEAT   EQU   *
         LA    0,12                           TEST FOR 12 MORE BYTES IN THE BUFFER
         BAL   14,POINT
         LA    1,RUNLIST                      OBTAIN TIME OF DAY, RUNUSER, NUMDIAL
         LA    2,3
```

(CONTINUED)

```
         DIAG  1,3
         L     10,0(3)
         LR    9,10
         USING UTABLE,9
AGAIN    EQU   *
         LA    0,USERID
         LA    1,NEXTUSER              OBTAIN ID OF NEXT USER
         STM   0,1,UTABLIST
         LA    1,UTABLIST
         LA    2,2
         LA    3,TEMPLIST
         DIAG  1,3
         CLC   TEMPLIST(3),=C'APL'     IS IT APL
         BE    BEGIN                   YES - BRANCH TO BEGIN
         L     9,TEMPLIST+4            NO - GET NEXT USER'S UTABLE
         CR    9,10
         BNE   AGAIN                   BRANCH TO AGAIN
BEGIN    LA    0,20
         BAL   14,POINT                TEST FOR 20 MORE BYTES IN THE BUFFER
         LA    4,TIMINQ
         LA    5,TIMEUSED
         LA    6,VTOTTIME
         LA    7,NUMPAGES
         LA    8,VMACHNO
         STM   4,8,APLLIST
         LA    1,APLLIST
         LA    2,5
         DIAG  1,3                     OBTAIN APL VIRTUAL MACHINE PARAMETERS
         LA    0,4                     TEST FOR 4 MORE BYTES IN THE BUFFER
         BAL   14,POINT
         MVC   0(4,R3),=C'EOIN'        INSERT END OF INFORMATION TAG
         LA    1,TYPEPL
         SVC   X'CA'                   PERFORM CONSOLE FUNCTION
         DC    AL4(*+4)
         B     AROUND
TYPEPL   DS    0D
         DC    CL8'TYPE'
         DC    AL1(1)
```

(CONTINUED)

```
         DC    AL3(*+7)                                       SEQUENCE OF 3 'SPACE-BACKSPACE'
         DC    C'K'
         DC    AL3(6)
         DC    3XL2'4016'
         DS    OH
AROUND   LA    1,WAITPL
         SVC   X'CA'
         DC    AL4(*+4)
         B     REPEAT                                         BRANCH TO REPEAT
WAITPL   DS    OD
         DC    CL8'WAIT'
         DC    CL4'CON1'
         DC    F'O'
         DS    1F
FCB      MADDPL NAME=STATS,TYPE=DATA,BUFF=BUFFER,SIZE=800
POINT    L     2,SAVENEXT
         LR    3,2                                            POINTER TO SPACE IN THE BUFFER IN REGISTER 2
         AR    2,0
         ST    2,SAVENEXT                                     TEST IF BUFFER IS FULL
         C     2,ENDBUFR
         BCR   4,14                                           NO - CONTINUE
         MVC   0(4,R3),=C'EOBB'                               YES - INSERT END OF BLOCK TAG
         WRBUF FCB,ERROR=PUTERR                               WRITE OUT ON TO DISK FILE
         L     7,COUNT
         LA    7,1(7)
         ST    7,COUNT                                        TEST IF 1 SAMPLE FINISHED
         C     7,LIMIT
         BC    10,QUIT                                        YES - QUIT
         LA    2,BUFFER
         LR    3,2                                            NO - RESET POINTER IN BUFFER
         AR    2,0
         ST    2,SAVENEXT
         BR    14
PUTERR   LA    15,8
         B     EXIT
QUIT     SR    15,15
EXIT     L     14,SAVE14
```

(CONTINUED)

```
         BR    14
SAVE14   DS    F
RUNLIST  DC    A(RUNUSER)
         DC    A(HOURS)
         DC    A(NUMDIAL)
UTABLIST DS    OD
         DS    A
         DS    A
TEMPLIST DS    OF
         DS    F
         DS    F
SAVENEXT DC    A(BUFFER)
ENDBUFR  DC    A(BUFEND)
BUFFER   DS    CL800
BUFEND   EQU   *
         DS    CL4
TEMPSAVE EQU   X'160'
RUNUSER  EQU   TEMPSAVE+64
HOURS    EQU   RUNUSER+8
NUMDIAL  EQU   X'916C'
APLLIST  DS    OD
         DS    A
         DS    A
         DS    A
         DS    A
         DS    A
COUNT    DC    F'1'
LIMIT    DC    F'101'
         COPY  UTABLE
         END
```

IV) ANAPL

```
ANAPL    START
         LR    12,15
```

(CONTINUED)

```
         USING ANAPL,12
         ST    14,SAVE14
         CMSREG
         STATE SFCB,NOFILE
         SETUP FCB
         RDBUF FCB,ERROR=INITERR                    READ FIRST 800 BYTES INTO THE BUFFER
         LA    3,BUFFER
NOW      EQU   *
         MVI   CARD,C'0'
         MVC   CARD+1(79),CARD                      SET POINTER IN THE BUFFER TO 4
         LA    0,4
         BAL   14,TAKE
         UNPK  CARD(7),4(4,R3)                      TIME OF DAY IN COL. 1 TO 7
         L     10,8(R3)
         SRA   10,16
         CVD   10,TEMP
         OI    TEMP+7,X'OF'
         UNPK  CARD+6(3),TEMP+6(2)                  NUMDIAL IN COL. 8 TO 10
         LA    0,12                                 MOVE POINTER 12 BYTES FORWARD
         BAL   14,TAKE
         L     11,4(R3)
         SRA   11,8
         M     10,CONST
         D     10,DIVISOR
         CVD   11,TEMP
         OI    TEMP+7,X'OF'
         UNPK  CARD+9(7),TEMP+4(4)                  TIMEUSED IN COL. 11 TO 17
         L     11,8(R3)
         SRA   11,8
         M     10,CONST
         D     10,DIVISOR
         CVD   11,TEMP
         OI    TEMP+7,X'OF'
         UNPK  CARD+16(7),TEMP+4(4)                 VTOTIME IN COL. 18 TO 24
         L     10,12(R3)
         SRA   10,16
         CVD   10,TEMP
```

(CONTINUED)

```
         OI    TEMP+7,X'OF'
         UNPK  CARD+23(7),TEMP+4(4)         NUMPAGES IN COL. 25 TO 31
         L     10,12(R3)
         SLA   10,16
         CVD   10,TEMP
         OI    TEMP+7,X'OF'
         UNPK  CARD+30(7),TEMP+4(4)         PRIORIT IN COL. 32 TO 38
         L     10,0(R3)
         SRA   10,8                         TEST IF TIMINQ=0
         BC    8,CHECK                      YES - BRANCH TO CHECK
         TM    3(R3),X'80'                  NO - TEST IF CONQBIT=0
         BC    8,Q2
         MVC   CARD+40(3),=C'001'
         B     MORE                         NO - INSERT 1 IN COL. 43 FOR Q1
Q2       MVC   CARD+37(3),=C'001'           YES - INSERT 1 IN COL. 40 FOR Q2
         B     MORE
CHECK    TM    3(R3),X'80'                  TEST IF CONQBIT=0
         BC    8,QUE2                       YES - BRANCH TO QUE2
         MVC   CARD+46(3),=C'001'           NO - INSERT 1 IN COL. 49 FOR EQ1
         B     MORE
QUE2     MVC   CARD+43(3),=C'001'           YES = INSERT 1 IN COL. 46 FOR EQ2
MORE     TM    18(R3),X'80'                 CHECK APL WAIT STATES
         BZ    *+10
         MVC   CARD+49(3),=C'001'           1 IN COL. 52 FOR PGWT
         TM    18(R3), X'40'
         BZ    *+10
         MVC   CARD+52(3),=C'001'           1 IN COL. 55 FOR IOWT
         TM    18(R3),X'20'
         BZ    *+10
         MVC   CARD+55(3),=C'001'           1 IN COL. 58 FOR CFWT
         TM    18(R3),X'04'
         BZ    *+10
         MVC   CARD+58(3),=C'001'           1 IN COL. 61 FOR EXCFN
         LA    1,PLIST
         SVC   X'CA'                        PUNCH A CARD
         LTR   15,15
         BNZ   PCHERR
```

(CONTINUED)

```
         LA     0,20
         BAL    14,TAKE
         B      NOW                                       BRANCH TO NOW
PCHERR   LA     15,4
         B      EXIT
EXIT     L      14,SAVE14
         BR     14
FCB      MADDPL NAME=STATS,TYPE=DATA,BUFF=BUFFER,SIZE=800
TAKE     AR     3,0
         CLC    0(4,R3),=C'EOBB'
         BCR    7,14
         RDBUF FCB,ERROR=GETERR
         LA     3,BUFFER
         BR     14
GETERR   CKEOF REALERR
         LA     15,10
         B      EXIT
REALERR  LA     15,8
         B      EXIT
NOFILE   LA     15,1
         B      EXIT
INITERR  LA     15,12
         B      EXIT
SFCB     DS     OD
         DC     CL8'STATE'
         DC     CL8'STATS'
         DC     CL8'DATA'
         DC     CL2'P1'
         DC     H'O'
         DC     E'O'
PLIST    DS     OD
         DC     CL8'CARDPH'
         DC     A(CARD)
SAVE14   DS     F
BUFFER   DS     CL800
TEMP     DS     D
```

(CONTINUED)

```
CARD     DS    CL80
CONST    DC    F'1000'
DIVISOR  DC    F'300'
         END
```

APPENDIX C

The series of tables, Table I(a) to Table I(n),gives the sample statistics of each variable in every sample. The number of observations varies between 1000 and 1200 observations. The series of tables, Table II(a) to Table II(g), gives the mean values of these variables for different values of the number of users. Each value is from a subsample with greater than 100 observations. The series of tables, Table III(a) to Table III(d),gives the sample statistics for the APL variables. The number of observations varies from approximately 550 to 700 observations. The APL virtual machine states are tabulated in Table III(e).

TABLE I(a)

NUMUSERS: THE NUMBER OF VIRTUAL MACHINES LOGGED ON

Sample Number	Mean	Standard Deviation	Maximum	Minimum
1	8.45	0.78	10.00	8.00
2	8.46	0.70	12.00	8.00
3	10.20	0.45	11.00	9.00
4	7.12	0.84	9.00	5.00
5	7.35	0.69	10.00	7.00
6	9.92	0.40	12.00	9.00
7	4.48	1.21	7.00	3.00
8	8.03	0.19	9.00	8.00
9	8.28	1.95	15.00	6.00
10	8.35	0.66	9.00	7.00
11	12.45	0.59	15.00	11.00
12	7.60	0.52	9.00	7.00
13	6.15	0.36	7.00	6.00
14	10.63	0.65	12.00	9.00
15	3.00	0.08	4.00	3.00
16	12.85	0.41	14.00	12.00

Overall Average = 8.3 users

TABLE I(b)

CPTIME: THE PERCENTAGE OF TIME THE SYSTEM WAS IN CP MODE

Sample Number	Mean	Standard Deviation	Maximum	Minimum
1	26.34	13.38	83.98	3.00
2	15.35	9.02	67.00	1.00
3	20.89	9.67	51.98	3.49
4	21.89	13.30	65.01	1.50
5	15.36	8.45	58.98	1.50
6	14.20	9.64	67.02	1.00
7	13.63	6.84	43.02	1.50
8	22.70	9.02	53.00	4.00
9	9.02	9.24	75.00	1.00
10	16.10	8.23	46.02	3.49
11	25.58	14.89	87.00	2.00
12	21.26	9.46	74.00	3.00
13	7.12	4.83	51.00	1.00
14	21.64	10.51	80.98	3.00
15	12.17	6.45	53.00	2.00
16	25.18	11.49	100.00	4.50

Overall Average = 18.03% of total time

TABLE I(c)

WAITTIME: THE PERCENTAGE OF TIME THE SYSTEM WAS IN "IDLE" STATE

Sample Number	Mean	Standard Deviation	Maximum	Minimum
1	4.04	12.21	100.00	0.00
2	69.33	32.91	100.00	0.00
3	0.53	2.77	60.01	0.00
4	46.21	36.07	100.00	0.00
5	14.52	29.85	100.00	0.00
6	53.91	41.50	100.00	0.00
7	18.26	33.75	100.00	0.00
8	1.79	3.98	33.98	0.00
9	85.66	24.58	100.00	0.00
10	2.87	13.18	100.00	0.00
11	41.12	36.25	100.00	0.00
12	6.95	17.84	100.00	0.00
13	63.16	44.00	100.00	0.00
14	7.58	21.27	100.00	0.00
15	11.70	28.16	100.00	0.00
16	4.58	7.47	43.02	0.00

Overall Average = 27.00 % of total time

TABLE I(d)

OVERHEAD: THE PERCENTAGE OF TIME THE SYSTEM SPENT IN OVERHEAD

Sample Number	Mean	Standard Deviation	Maximum	Minimum
1	9.79	5.09	25.98	1.00
2	4.24	2.47	16.00	0.00
3	7.30	3.34	20.00	1.50
4	6.77	4.88	27.00	0.00
5	5.46	2.86	18.01	0.00
6	4.22	2.66	17.00	0.50
7	5.86	3.09	17.02	0.00
8	7.56	2.87	20.00	1.00
9	2.14	1.98	18.00	0.00
10	5.81	2.77	17.99	1.00
11	7.25	4.25	30.00	0.00
12	7.87	1.11	23.00	1.00
13	2.64	1.81	16.00	0.00
14	6.86	3.12	26.00	0.50
15	4.87	2.56	17.02	0.00
16	7.14	3.09	26.00	1.50

Overall Average = 6.00 % of total time

TABLE I(e)

PROBTIME: THE PERCENTAGE OF TIME THE SYSTEM WAS IN PROBLEM MODE

Sample Number	Mean	Standard Deviation	Maximum	Minimum
1	69.60	19.43	97.00	0.00
2	15.30	31.03	97.00	0.00
3	78.56	10.70	96.51	8.98
4	31.88	32.83	95.00	0.00
5	70.10	29.81	97.00	0.00
6	31.88	39.70	97.00	0.00
7	68.09	33.53	97.01	0.00
8	75.49	11.27	96.00	0.00
9	5.30	20.91	95.00	0.00
10	81.00	15.91	96.51	0.00
11	33.28	35.42	96.00	0.00
12	71.76	19.93	96.51	0.00
13	29.70	42.75	97.50	0.00
14	70.76	22.54	96.00	0.00
15	76.11	28.24	97.51	0.00
16	70.22	14.76	95.50	0.00

Overall Average = 54.96% of total time

TABLE I(f)

PGRD: THE NUMBER OF PAGES READ IN PER SECOND

Sample Number	Mean	Standard Deviation	Total in 20 Minutes	Maximum	Minimum
1	0.70	1.91	747.82	18.00	0.00
2	0.57	1.68	610.50	20.00	0.00
3	1.88	3.51	2,052.82	33.00	0.00
4	0.66	1.92	716.50	16.00	0.00
5	0.74	2.13	808.50	26.00	0.00
6	1.18	2.88	1,160.83	35.00	0.00
7	0.06	0.50	68.50	9.00	0.00
8	1.52	3.38	1,685.16	40.00	0.00
9	0.77	2.34	899.50	27.00	0.00
10	0.40	1.31	424.50	13.00	0.00
11	3.42	5.62	3,840.66	48.00	0.00
12	0.34	1.11	407.50	11.00	0.00
13	0.44	1.42	491.83	11.00	0.00
14	2.42	4.65	2,674.50	49.00	0.00
15	0.01	0.12	15.00	2.00	0.00
16	4.47	5.48	4,803.50	45.00	0.00

Overall Average: 1.1 pages per second

TABLE I(g)

PGSWP: THE NUMBER OF PAGES SWAPPED (READ OUT) PER SECOND

Sample Number	Mean	Standard Deviation	Total in 20 Minutes	Maximum	Minimum
1	0.56	1.55	598.16	15.00	0.00
2	0.51	1.59	547.00	19.00	0.00
3	1.69	3.15	1,842.82	30.00	0.00
4	0.42	1.49	453.00	16.00	0.00
5	0.64	1.87	698.50	25.00	0.00
6	1.00	2.50	985.33	28.00	0.00
7	0.03	0.34	33.50	9.00	0.00
8	1.32	2.98	1,458.83	37.00	0.00
9	0.59	1.99	689.50	24.00	0.00
10	0.34	1.19	363.00	12.00	0.00
11	2.81	4.77	3,156.16	36.00	0.00
12	0.29	0.98	354.50	10.00	0.00
13	0.38	1.27	433.83	10.00	0.00
14	2.06	3.96	2,285.00	38.00	0.00
15	0.00	0.00	0.00	0.00	0.00
16	3.94	4.87	4,231.50	34.00	0.00

Overall Average: 1.04 pages per second

TABLE I(h)

PGWT: THE PERCENTAGE OF USERS IN PAGE WAIT

Sample Number	Mean	Standard Deviation	Maximum	Minimum
1	1.61	6.08	62.50	0.00
2	0.39	2.38	28.57	0.00
3	0.78	2.89	22.22	0.00
4	0.30	2.27	28.57	0.00
5	0.38	2.40	16.67	0.00
6	0.46	2.34	25.00	0.00
7	0.20	2.58	50.00	0.00
8	0.89	3.56	28.57	0.00
9	0.61	3.04	33.33	0.00
10	0.24	1.98	28.57	0.00
11	1.37	4.01	41.67	0.00
12	0.88	3.77	33.33	0.00
13	0.39	2.85	40.00	0.00
14	1.18	4.11	60.00	0.00
15	0.08	2.11	50.00	0.00
16	1.96	4.46	33.33	0.00

Overall Average = 0.8% of users

TABLE I(i)

IOWT: THE PERCENTAGE OF USERS IN I/O WAIT

Sample Number	Mean	Standard Deviation	Maximum	Minimum
1	5.30	8.59	42.86	0.00
2	1.78	4.70	25.00	0.00
3	3.21	6.26	40.00	0.00
4	4.06	8.17	42.86	0.00
5	1.74	5.16	37.50	0.00
6	0.98	3.30	22.22	0.00
7	1.66	7.54	50.00	0.00
8	5.26	9.28	42.86	0.00
9	0.91	3.67	28.57	0.00
10	1.73	4.79	28.57	0.00
11	2.06	4.60	27.27	0.00
12	3.49	7.32	42.86	0.00
13	0.44	2.92	20.00	0.00
14	3.00	6.32	50.00	0.00
15	2.85	11.80	100.00	0.00
16	2.78	5.26	25.00	0.00

Overall Average = 2.58% of users

TABLE I(j)

CFWT: THE PERCENTAGE OF USERS IN CONSOLE FUNCTION WAIT

Sample Number	Mean	Standard Deviation	Maximum	Minimum
1	12.51	11.45	42.86	0.00
2	5.17	7.29	36.36	0.00
3	2.88	5.35	22.22	0.00
4	18.60	16.97	71.43	0.00
5	1.37	4.16	22.22	0.00
6	3.80	6.02	22.22	0.00
7	3.82	9.39	40.00	0.00
8	2.40	6.02	28.57	0.00
9	9.06	13.79	66.67	0.00
10	8.00	8.90	28.57	0.00
11	4.04	5.43	25.00	0.00
12	5.92	7.68	37.50	0.00
13	2.34	6.21	33.33	0.00
14	8.61	8.53	40.00	0.00
15	0.26	2.97	33.33	0.00
16	2.81	5.02	16.67	0.00

Overall Average = 5.7% of users

TABLE I(k)

EQ1: THE PERCENTAGE OF USERS ELIGIBLE TO ENTER Q_1

Sample Number	Mean	Standard Deviation	Maximum	Minimum
1	64.69	14.12	100.00	12.50
2	91.74	9.90	100.00	57.14
3	64.80	11.70	90.00	33.33
4	81.20	15.86	100.00	40.00
5	76.57	11.62	100.00	33.33
6	89.26	9.19	100.00	50.00
7	71.85	16.69	100.00	0.00
8	62.91	12.86	100.00	28.57
9	90.39	10.57	100.00	50.00
10	76.02	10.83	100.00	16.67
11	86.62	11.08	100.00	25.00
12	75.18	12.78	100.00	33.33
13	86.47	11.01	100.00	40.00
14	73.87	14.91	100.00	10.00
15	50.96	20.13	100.00	0.00
16	76.54	8.33	100.00	41.67

Overall Average = 76.32% of users

TABLE I(1)

Q1: THE PERCENTAGE OF USERS IN Q_1

Sample Number	Mean	Standard Deviation	Maximum	Minimum
1	3.61	7.70	55.56	0.00
2	4.61	7.59	42.86	0.00
3	2.77	5.51	33.33	0.00
4	6.77	9.56	50.00	0.00
5	3.34	6.78	37.50	0.00
6	5.38	6.91	37.50	0.00
7	1.49	7.09	100.00	0.00
8	4.63	7.75	42.86	0.00
9	7.72	8.79	37.50	0.00
10	1.59	4.82	50.00	0.00
11	5.39	6.69	41.67	0.00
12	3.86	7.61	57.14	0.00
13	6.29	9.51	40.00	0.00
14	4.31	6.97	40.00	0.00
15	2.76	11.43	50.00	0.00
16	6.18	7.05	41.67	0.00

Overall Average = 4.42% of users

TABLE I(m)

EQ2: THE PERCENTAGE OF USERS ELIGIBLE TO ENTER Q_2

Sample Number	Mean	Standard Deviation	Maximum	Minimum
1	2.44	5.34	28.57	0.00
2	0.50	2.60	14.29	0.00
3	1.79	4.13	22.22	0.00
4	0.99	3.99	20.00	0.00
5	2.24	5.71	33.33	0.00
6	0.71	2.72	12.50	0.00
7	5.59	12.89	50.00	0.00
8	2.50	5.53	28.57	0.00
9	0.12	1.33	16.67	0.00
10	2.13	5.34	33.33	0.00
11	0.75	2.48	16.67	0.00
12	2.47	5.68	28.57	0.00
13	0.76	3.83	20.00	0.00
14	1.45	3.67	20.00	0.00
15	7.51	18.22	100.00	0.00
16	1.18	3.00	18.18	0.00

Overall Average = 2.08% of users

TABLE I(n)

Q2: THE PERCENTAGE OF USERS IN Q_2

Sample Number	Mean	Standard Deviation	Maximum	Minimum
1	29.24	14.17	71.43	0.00
2	3.13	6.33	28.57	0.00
3	30.62	12.01	55.56	0.00
4	11.01	11.82	57.14	0.00
5	17.83	11.79	50.00	0.00
6	4.62	6.24	33.33	0.00
7	21.05	18.39	66.67	0.00
8	29.93	12.23	57.14	0.00
9	1.75	4.74	25.00	0.00
10	20.24	10.93	50.00	0.00
11	7.21	8.32	41.67	0.00
12	18.46	11.85	50.00	0.00
13	6.46	9.64	40.00	0.00
14	20.35	13.93	60.00	0.00
15	38.75	24.83	100.00	0.00
16	16.08	6.95	36.36	0.00

Overall Average = 17.3% of users

TABLE II(a)

THE MEAN VALUES OF THE FIFTEEN VARIABLES FOR THREE USERS

CPTIME	WAITTIME	OVERHEAD	PROBTIME	PGRD	PGSWP	PGWT	IOWT	CFWT	Q1	Q2	EQ1	EQ2
13.65	2.39	6.19	83.96	0.00	0.00	0.00	2.86	0.00	0.00	40.10	50.78	9.11
14.70	6.05	5.55	79.24	0.06	0.00	0.44	6.77	0.00	3.49	46.50	41.70	8.30
11.77	10.53	4.74	77.69	0.00	0.00	0.00	2.00	0.00	2.67	38.25	51.67	7.41
9.84	27.81	4.32	62.35	0.00	0.00	0.00	0.78	0.00	2.34	29.69	61.33	6.64

TABLE II(b)

THE MEAN VALUES OF THE FIFTEEN VARIABLES FOR FOUR USERS

CPTIME	WAITTIME	OVERHEAD	PROBTIME	PGRD	PGSWP	PGWT	IOWT	CFWT	Q1	Q2	EQ1	EQ2
13.71	5.77	6.07	80.52	0.09	0.06	0.28	1.60	1.60	1.41	25.70	66.29	6.59

TABLE II(c)

THE MEAN VALUES OF THE FIFTEEN VARIABLES FOR SEVEN USERS

CPTIME	WAITTIME	OVERHEAD	PROBTIME	PGRD	PGSWP	PGWT	IOWT	CFWT	Q1	Q2	EQ1	EQ2
17.32	37.71	5.73	44.97	0.23	0.06	0.15	2.40	33.33	4.50	12.76	80.93	1.80
17.79	48.13	5.48	34.07	0.51	0.47	0.25	3.88	9.40	5.77	10.03	83.83	0.38
13.67	8.56	5.00	77.78	0.49	0.36	0.29	1.37	0.04	2.83	24.65	70.45	2.08
15.44	14.50	5.88	70.06	0.39	0.36	0.23	2.00	0.00	2.48	14.19	80.61	2.72
3.75	97.03	1.00	0.00	0.03	0.00	0.00	0.32	0.97	12.79	0.00	87.21	0.00
21.97	4.50	8.30	73.53	0.25	0.21	0.76	3.66	4.66	4.45	22.81	69.91	2.83

TABLE II(d)

THE MEAN VALUES OF THE FIFTEEN VARIABLES FOR EIGHT USERS

CPTIME	WAITTIME	OVERHEAD	PROBTIME	PGRD	PGSWP	PGWT	IOWT	CFWT	Q1	Q2	EQ1	EQ2
28.90	0.57	11.52	70.53	0.26	0.15	0.41	5.40	9.36	1.22	36.51	59.52	2.73
22.16	2.47	8.00	75.37	0.37	0.29	1.08	4.76	19.23	3.56	19.17	74.43	2.83
10.63	67.97	2.91	21.40	1.04	0.93	1.17	0.70	4.33	6.79	3.63	88.87	0.70
16.83	33.21	5.40	49.96	0.35	0.32	0.14	1.31	0.89	3.09	9.14	86.26	1.51
23.05	59.82	6.46	17.13	1.21	0.85	0.41	4.00	2.65	9.06	6.35	84.07	0.53
22.06	1.49	7.51	76.44	1.68	1.48	1.11	4.05	0.29	4.18	31.47	61.54	2.81
22.44	1.10	7.48	76.46	1.88	1.53	1.23	5.34	0.00	6.24	31.44	59.85	2.46
24.33	2.53	7.86	73.14	1.31	1.15	0.72	6.81	7.75	4.32	28.96	64.17	2.55
16.43	0.70	5.75	82.87	0.46	0.43	0.30	1.46	0.79	1.46	27.90	68.17	2.47
25.77	3.02	10.09	71.20	0.61	0.58	1.76	5.80	1.86	3.11	25.36	68.74	2.80
21.43	6.36	7.20	72.21	0.32	0.31	0.78	3.96	8.31	5.13	14.68	78.89	1.30
18.32	17.46	6.46	64.21	0.24	0.24	0.43	1.71	5.00	2.21	10.86	84.35	2.57

TABLE II(e)

THE MEAN VALUES OF THE FIFTEEN VARIABLES FOR TEN USERS

CPTIME	WAITTIME	OVERHEAD	PROBTIME	PGRD	PGSWP	PGWT	IOWT	CFWT	EXCFN	Q1	Q2	EQ1	EQ2
27.72	11.26	8.69	61.02	1.50	1.35	2.58	5.10	5.38	0.00	8.48	29.55	60.02	1.43
17.32	0.08	6.52	82.60	3.34	3.06	1.33	1.07	0.00	0.00	1.70	44.00	52.33	1.96
18.74	0.11	6.89	81.14	1.66	1.54	0.68	2.38	4.22	0.00	2.15	32.73	63.15	1.96
25.59	1.41	7.95	73.00	2.41	2.03	1.17	4.34	2.62	0.00	4.34	21.32	72.54	1.81
28.71	2.39	10.08	68.90	0.91	0.64	1.95	7.89	1.75	0.00	3.80	29.33	64.91	1.94
10.59	37.15	3.67	52.26	1.32	1.18	0.32	0.56	5.80	0.00	3.38	6.12	89.85	0.64
14.11	55.32	3.72	30.57	1.51	1.38	0.51	0.85	0.25	0.00	8.65	5.68	84.82	0.85
12.88	83.29	3.87	3.83	0.76	0.71	0.08	0.97	11.27	0.00	5.03	1.14	93.59	0.24
16.32	55.02	4.79	28.66	0.51	0.32	0.32	1.13	0.72	0.00	4.15	4.47	90.60	0.77

TABLE II(f)

THE MEAN VALUES OF THE FIFTEEN VARIABLES FOR ELEVEN USERS

CPTIME	WAITTIME	OVERHEAD	PROBTIME	PGRD	PGSWP	PGWT	IOWT	CFWT	EXCFN	Q1	Q2	EQ1	EQ2
19.49	0.15	6.26	80.36	1.54	1.42	0.71	3.05	3.22	0.00	3.43	22.97	72.38	1.21
23.66	2.20	7.16	74.14	1.77	1.54	0.81	2.43	11.68	0.00	3.40	17.21	77.92	1.47
19.15	14.57	6.05	66.28	2.25	2.02	1.04	1.50	13.96	0.00	7.37	10.04	81.29	1.29
23.76	0.53	7.43	75.71	4.66	3.91	2.45	3.94	2.74	0.00	3.91	38.83	56.06	1.20

TABLE II(g)

THE MEAN VALUES OF THE FIFTEEN VARIABLES FOR THIRTEEN USERS

CPTIME	WAITTIME	OVERHEAD	PROBTIME	PGRD	PGSWP	PGWT	IOWT	CFWT	EXCFN	Q1	Q2	EQ1	EQ2
17.84	37.56	5.58	44.60	2.25	1.67	1.01	1.09	8.84	0.00	4.13	5.80	88.77	1.30
35.62	17.12	11.06	47.26	8.02	6.80	3.62	3.32	4.83	0.42	6.52	14.73	78.01	0.72
32.38	7.06	8.48	60.56	6.97	6.22	2.61	4.58	8.52	0.00	7.92	17.92	73.41	0.76
23.89	4.92	6.90	71.19	3.71	3.15	1.65	1.34	0.17	0.00	6.51	12.09	80.06	1.34
19.45	1.50	6.39	79.04	1.94	1.81	1.13	1.53	0.68	0.00	3.11	14.12	81.63	1.13
23.69	5.96	6.39	70.34	4.55	3.91	2.64	3.25	1.57	0.05	8.38	15.75	74.80	1.07

TABLE III(a)

NUMDIAL: THE NUMBER OF USERS DIALLED ON TO THE APL VIRTUAL MACHINE

Sample Number	Mean	Standard Deviation	Maximum	Minimum
1	16.00	0.00	16.00	16.00
2	16.56	0.58	17.00	15.00
3	14.94	0.76	16.00	14.00
4	7.50	0.86	10.00	6.00
5	10.98	0.90	12.00	9.00
6	11.26	0.44	12.00	11.00

Overall Average = 12.87 users

TABLE III(b)

TIMEUSED: THE PERCENTAGE OF TIME THE APL MACHINE WAS IN PROBLEM MODE OR CP MODE

Sample Number	Mean	Standard Deviation	Maximum	Minimum
1	37.27	43.37	100.00	0.20
2	53.37	31.78	100.00	0.98
3	18.45	21.35	99.40	0.70
4	19.93	31.73	100.00	0.30
5	23.12	32.34	100.00	0.29
6	56.70	43.36	100.00	0.29

Overall Average = 34.8% of total time

TABLE III(c)

VTOTTIME: THE PERCENTAGE OF TIME THE APL MACHINE WAS IN PROBLEM MODE

Sample Number	Mean	Standard Deviation	Maximum	Minimum
1	33.94	43.01	100.00	0.00
2	45.96	32.22	100.00	0.00
3	10.51	20.19	98.40	0.00
4	16.95	31.23	100.00	0.00
5	19.65	31.68	100.00	0.00
6	53.58	43.18	100.00	0.00

Overall Average = 30.1% of total time

TABLE III(d)

NUMPAGES: THE NUMBER OF PAGES THE APL MACHINE HAD IN CORE

Sample Number	Mean	Standard Deviation	Maximum	Minimum
1	48.97	2.81	52.00	39.00
2	50.82	2.13	52.00	32.00
3	61.75	10.19	91.00	48.00
4	58.74	13.35	90.00	39.00
5	50.25	2.81	52.00	29.00
6	52.00	0.00	52.00	52.00

Overall Average = 53.7 pages

TABLE III(e)

THE PERCENTAGE OF TIME THE APL MACHINE WAS IN DIFFERENT STATES

Sample Number	Q2	Q1	EQ2	EQ1	PGWT	IOWT	CFWT	EXCFN
1	32.9	33.8	5.4	27.7	0.2	0.7	0.0	0.0
2	51.4	16.4	18.5	13.5	2.8	7.4	0.0	0.0
3	13.4	41.8	4.4	40.2	0.5	6.0	0.0	0.0
4	21.0	49.1	2.4	27.4	0.3	2.9	0.0	0.0
5	24.2	43.4	4.9	27.3	0.7	1.4	0.0	0.0
6	53.3	12.1	7.6	26.9	0.0	1.4	0.0	0.0
Overall Average	32.7%	32.8%	7.2%	27.2%	0.75%	3.3%	0.0%	0.0%

APPENDIX D

Histograms in Figure H1 to H13 represent the frequency plots of the variables CPTIME, WAITTIME (low value), WAITTIME (high value), OVERHEAD, PGREAD, PGSWAP, PGWT, IOWT, CFWT, EQ1, Q1, Q2, EQ2, and PROBTIME, respectively. Every tenth value of the variable was used. The sample output given here is from sample number 3. The high values of WAITTIME are plotted from sample number 9. Histograms in Figure H14 to H17 are the frequency plots of the "APL virtual machine" variables (TIMEUSED - VTOTTIME), TIMEUSED, VTOTTIME, and NUMPAGES. In this case, every fifth value of the variables is plotted. The output in this appendix is from sample number 4.

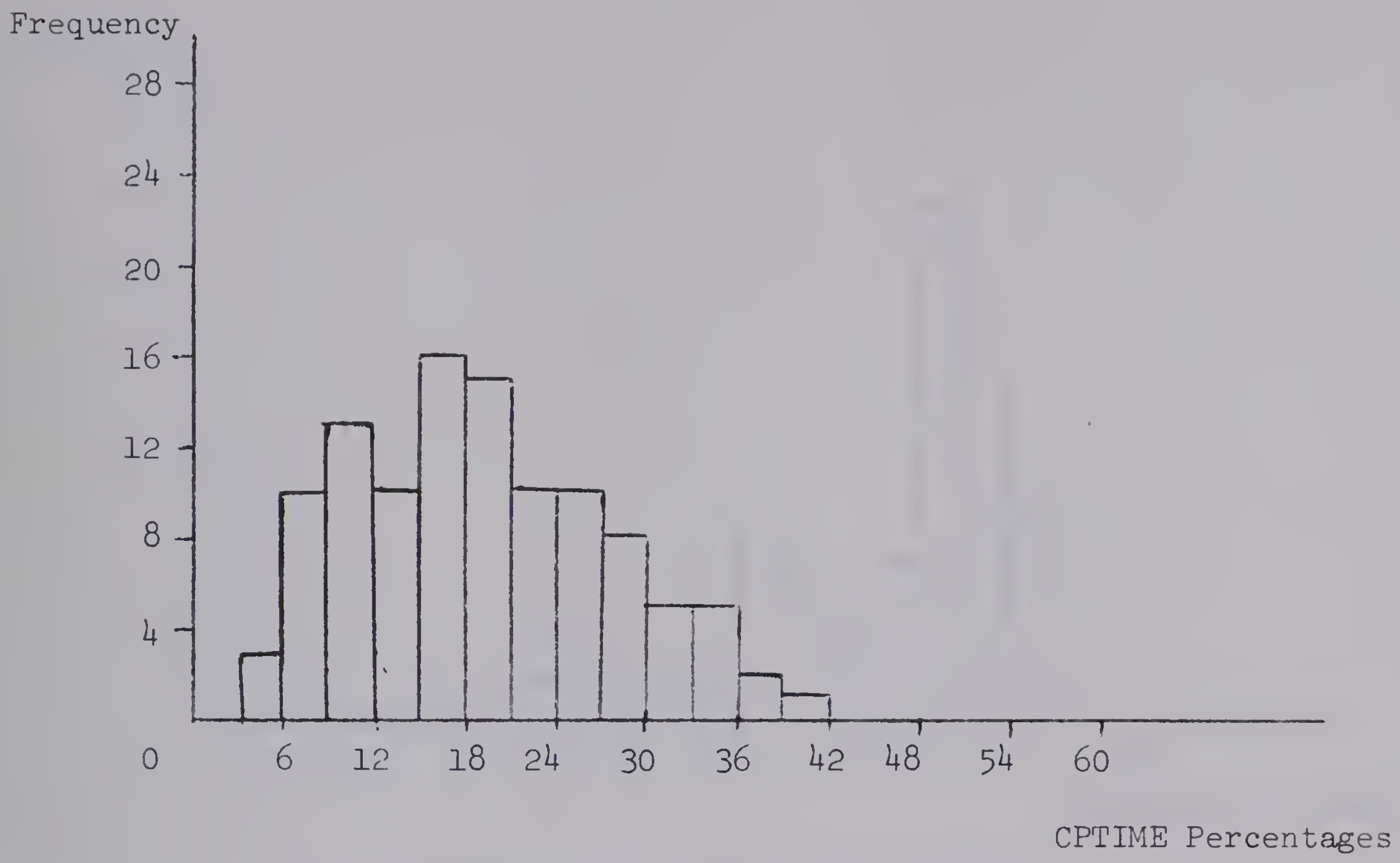

Fig. H1. Histogram of CPTIME

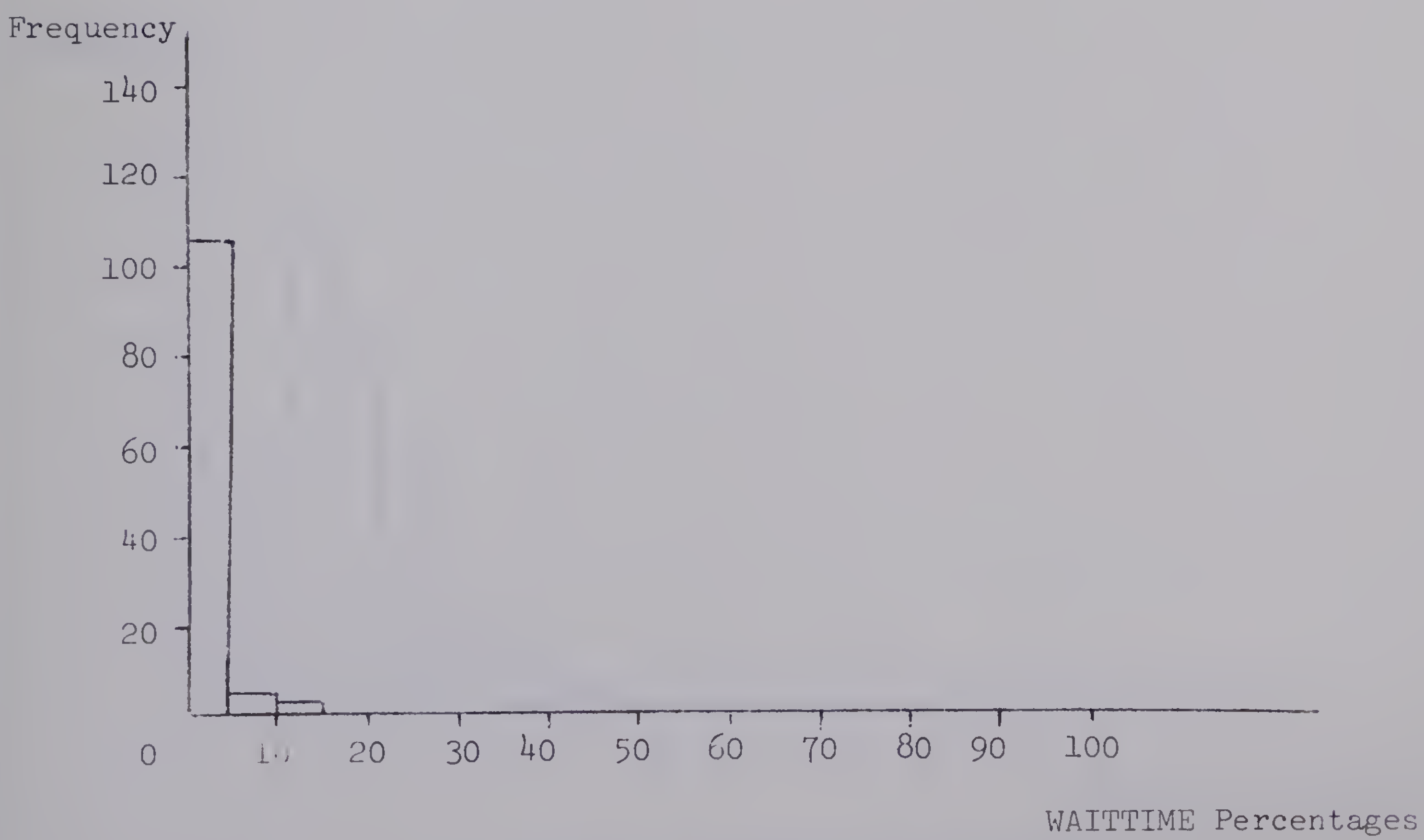

Fig. H2(a). Histogram of Low Values of WAITTIME

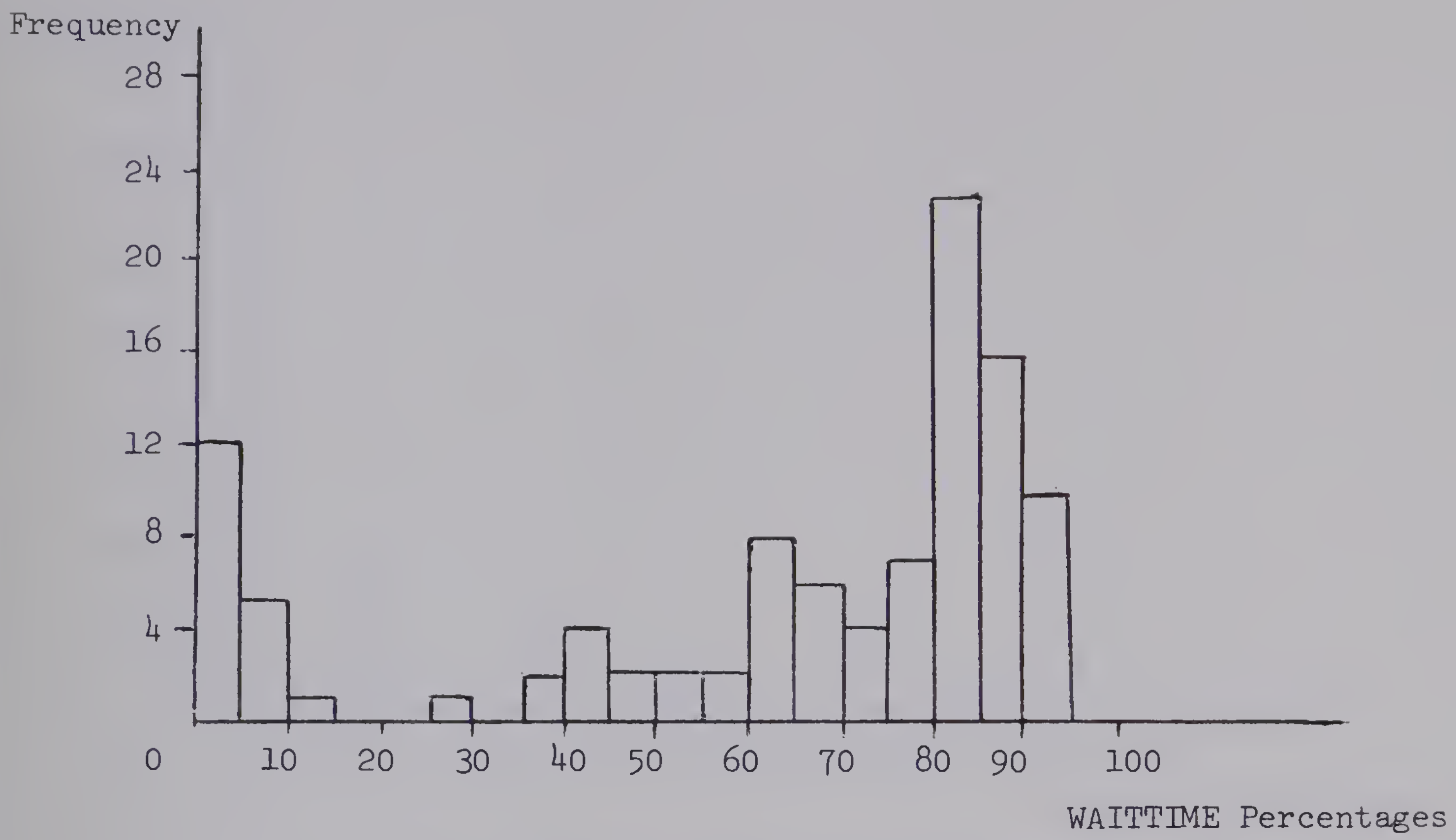

Fig. H2(b). Histogram of High Values of WAITTIME

Fig. H3. Histogram of OVERHEAD

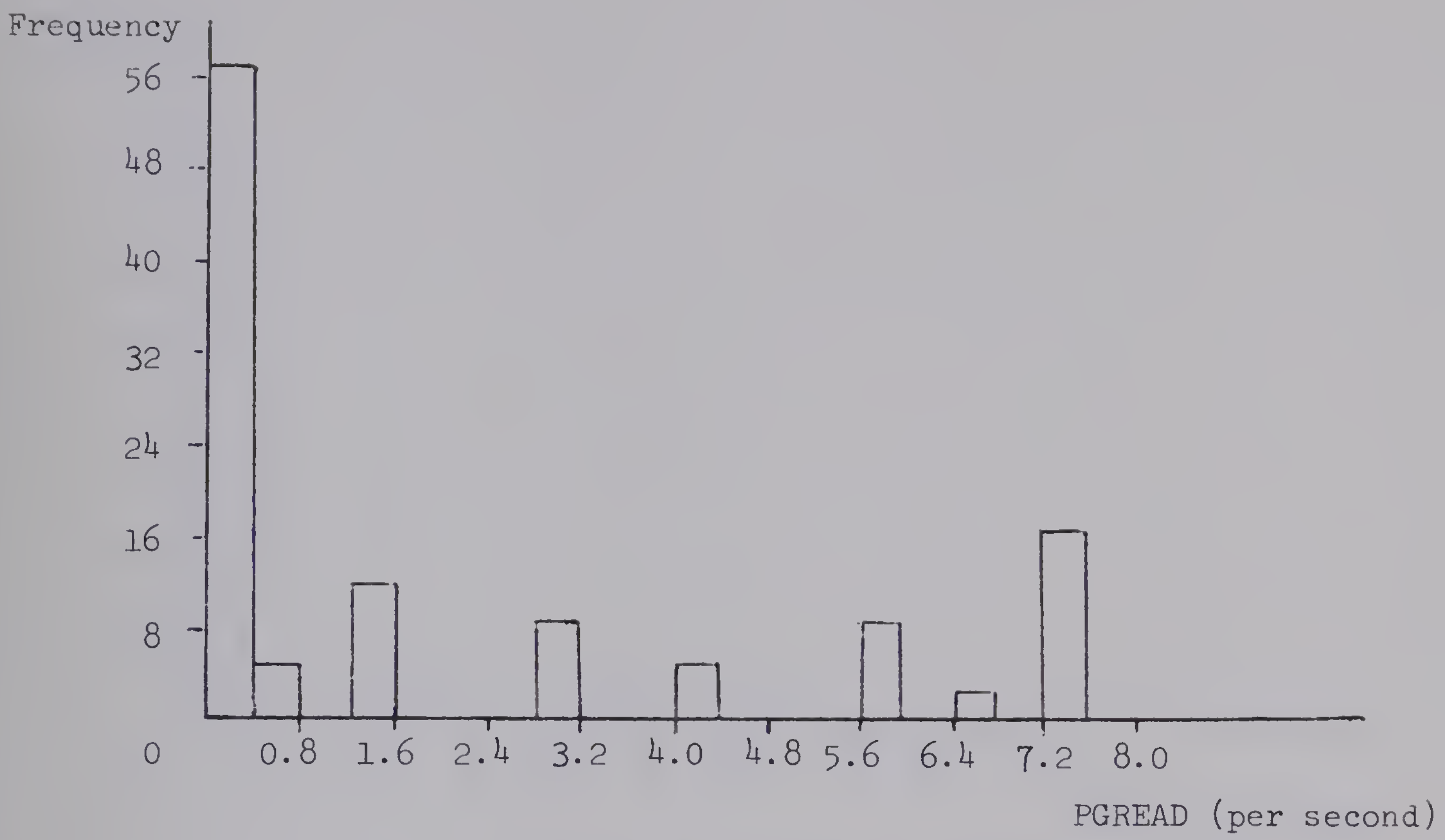

Fig. H4. Histogram of PGREAD

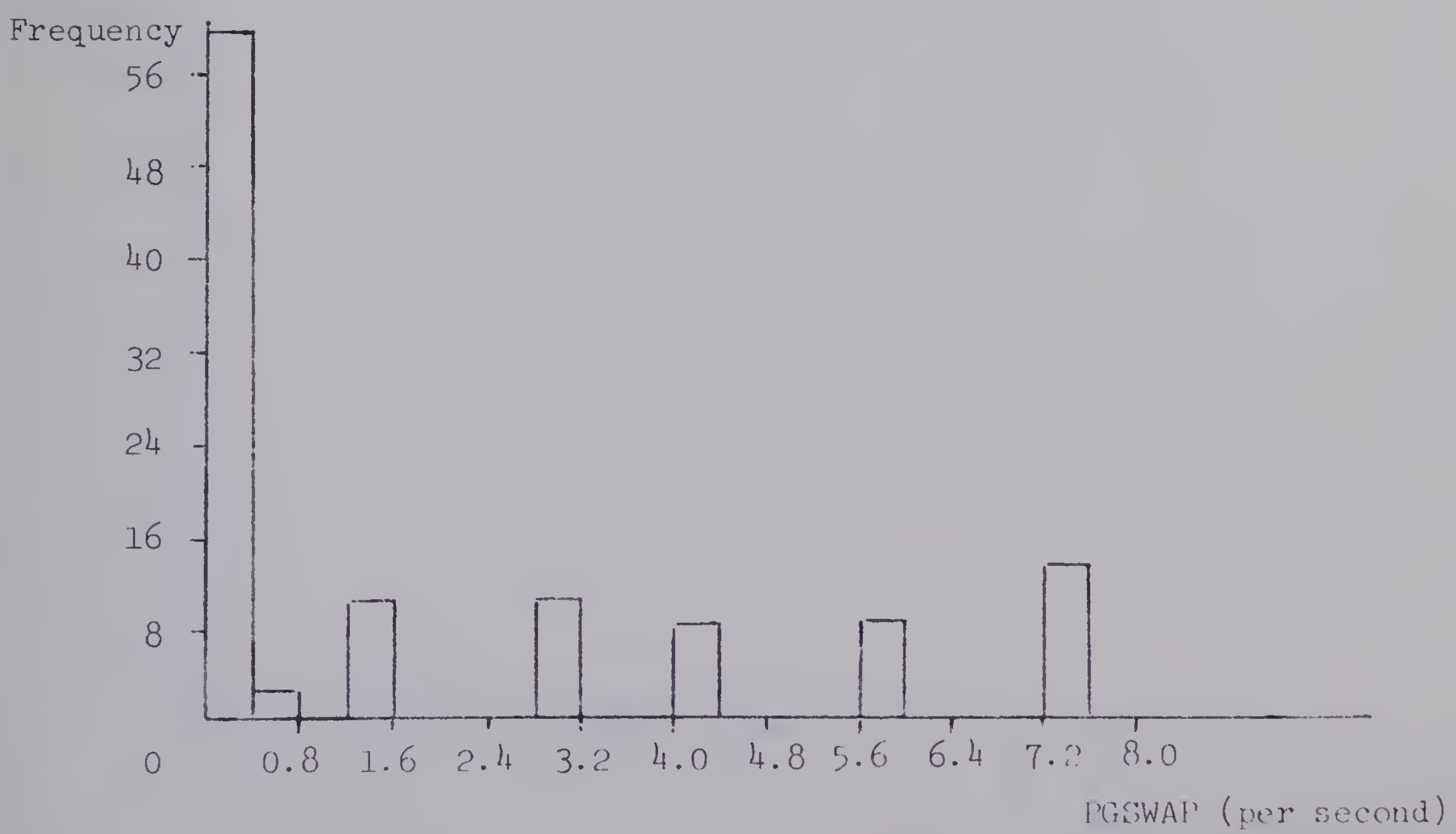

Fig. H5. Histogram of PGSWAP

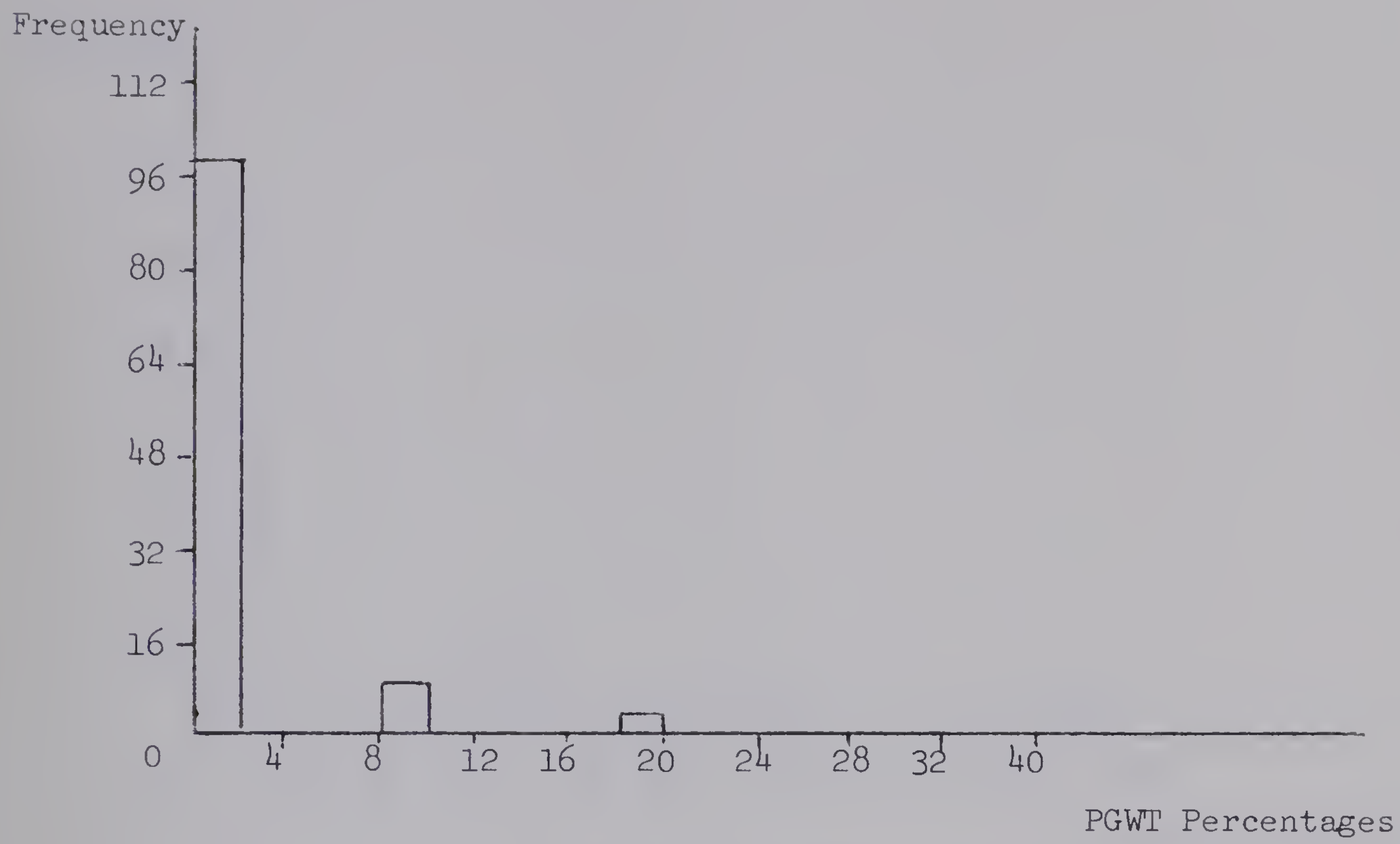

Fig. H6. Histogram of PGWT

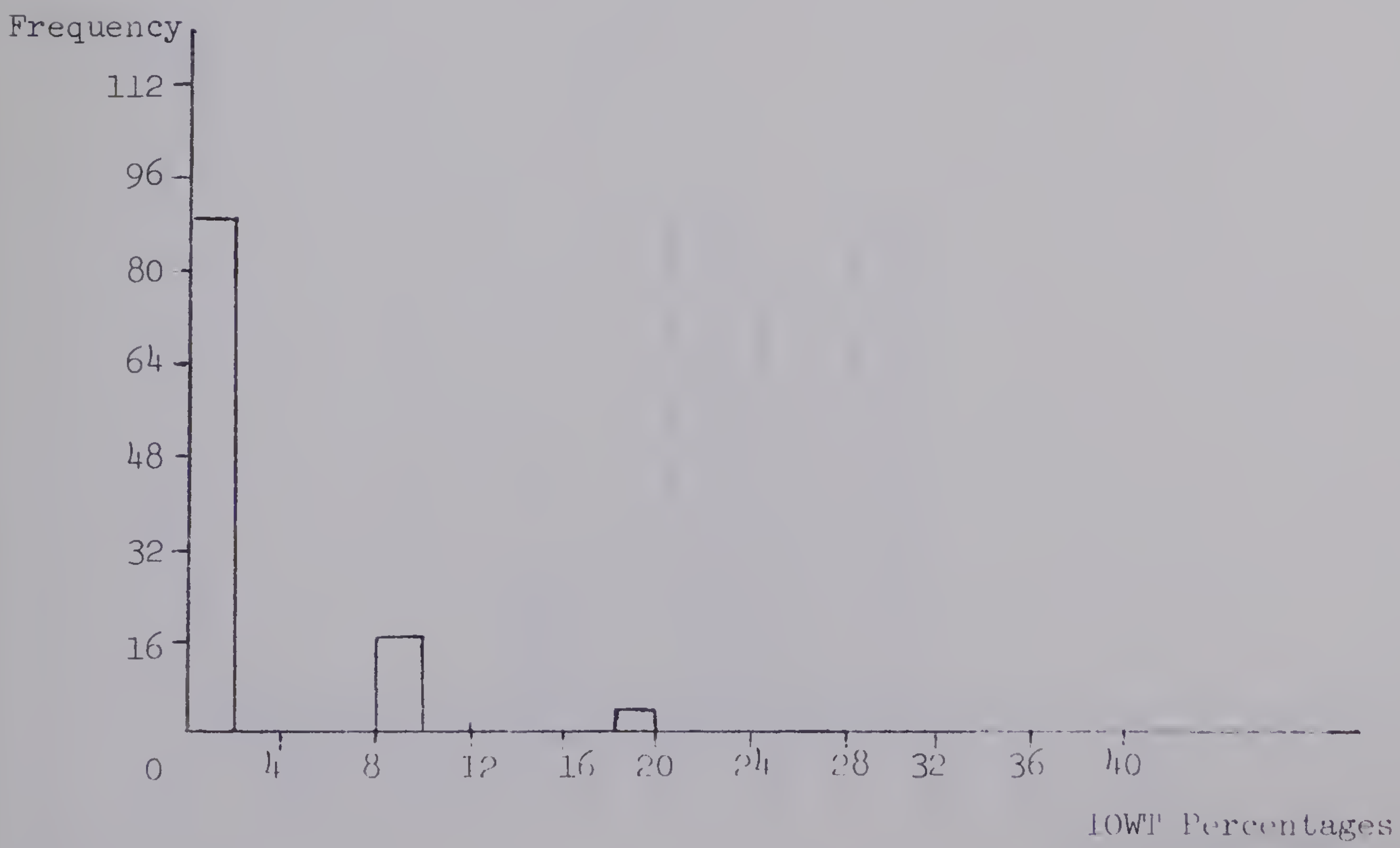

Fig. H7. Histogram of IOWT

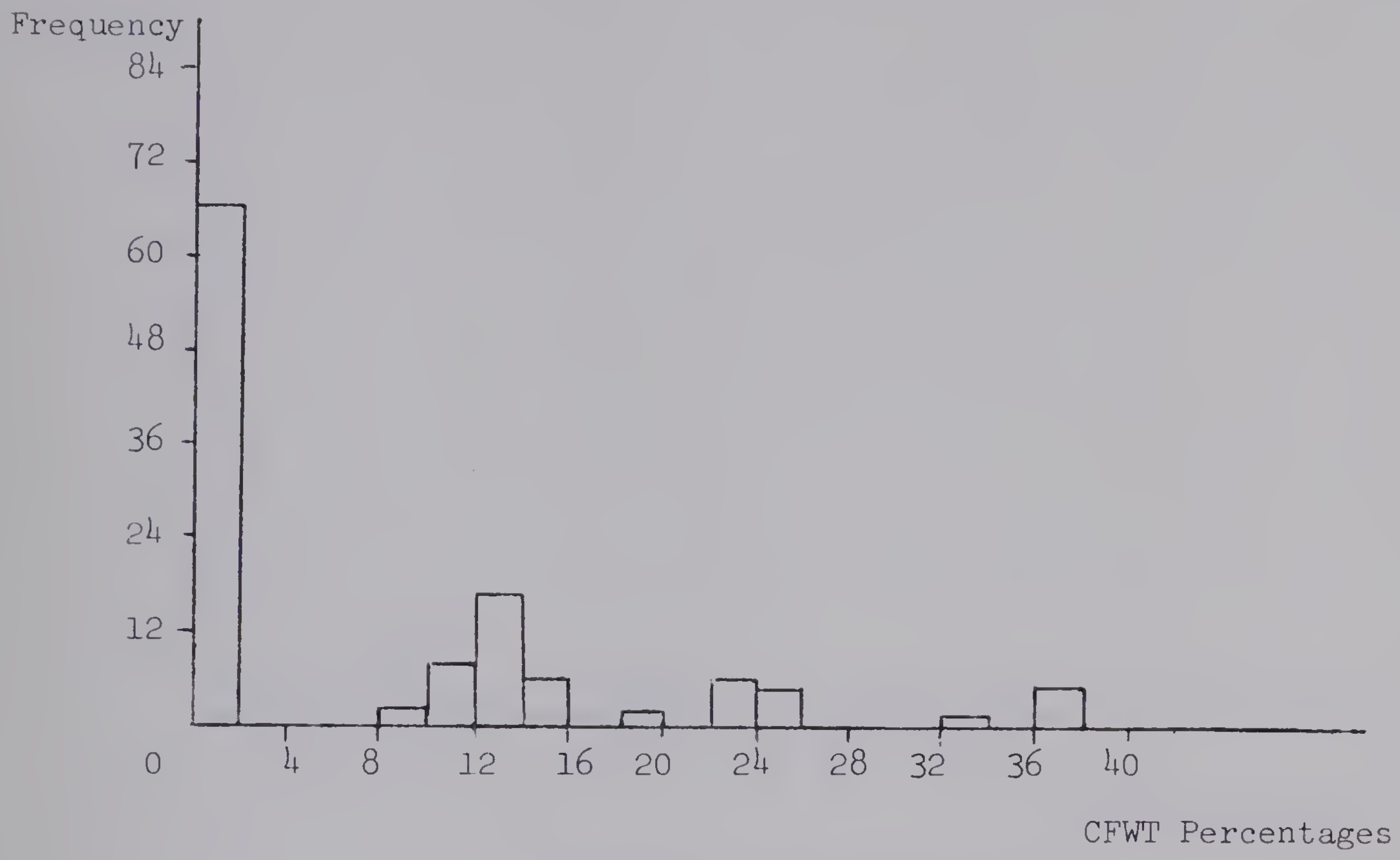

Fig. H8. Histogram of CFWT

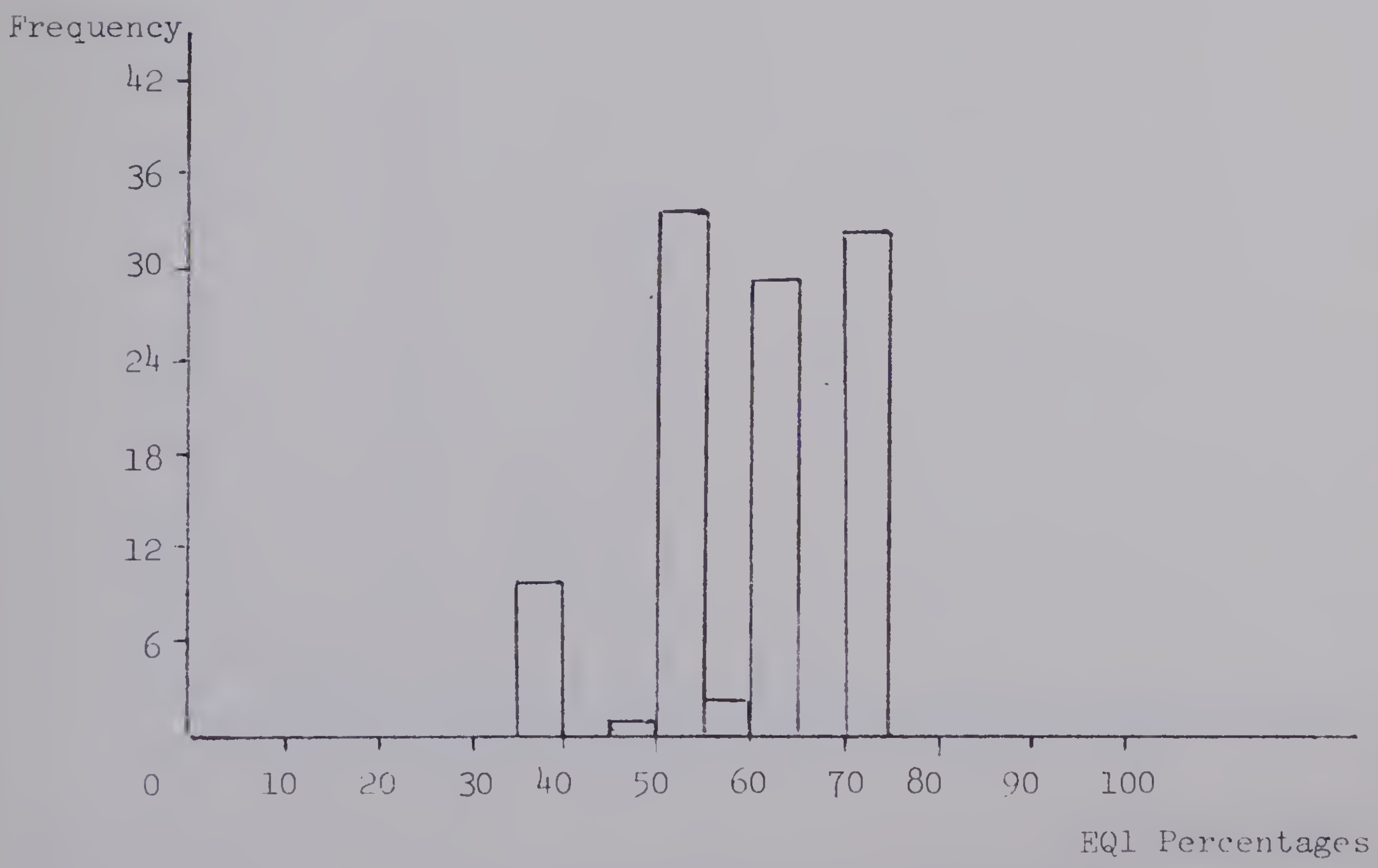

Fig. H9. Histogram of EQ1

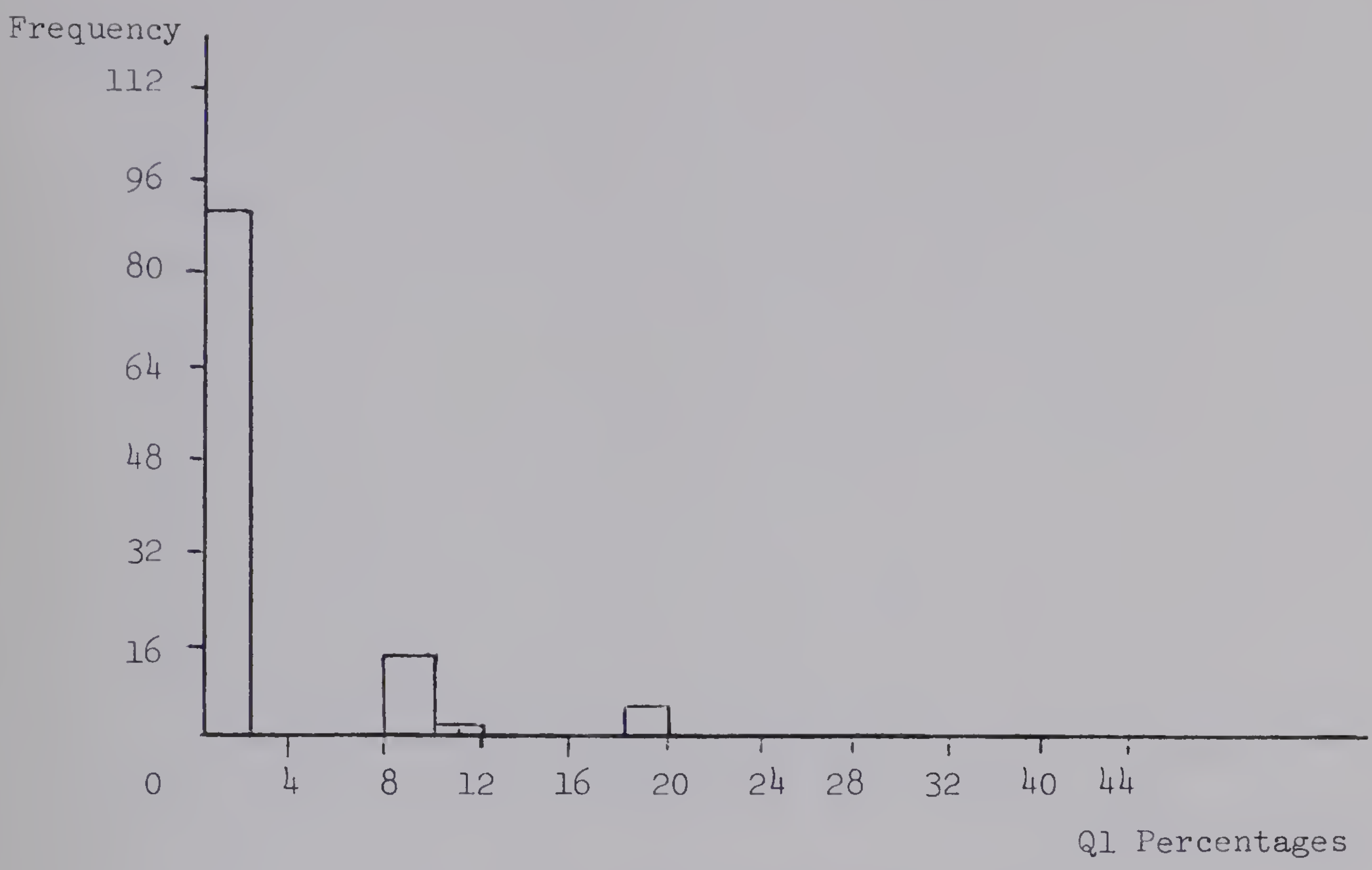

Fig. H10. Histogram of Q1

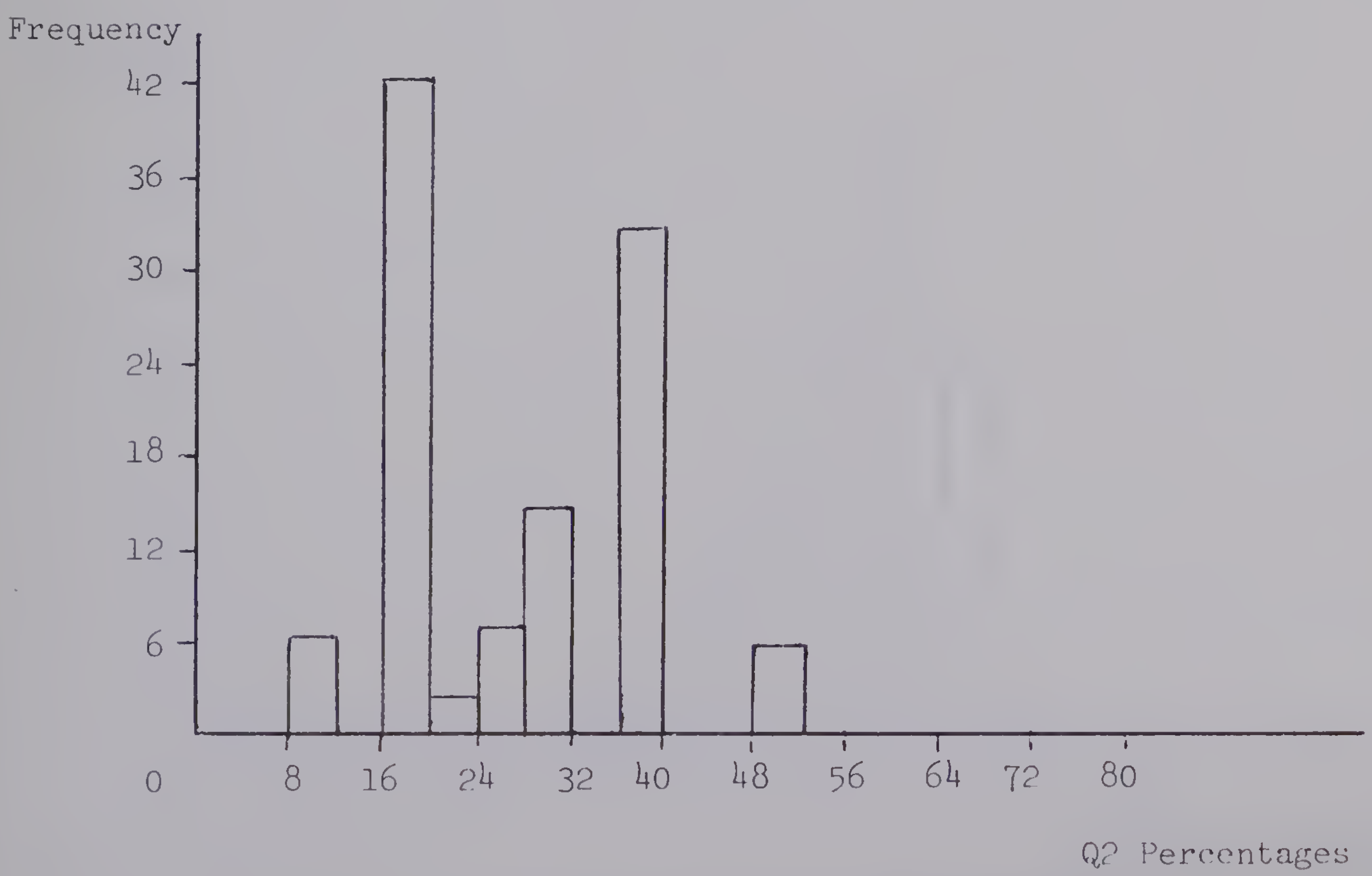

Fig. H11. Histogram of Q2

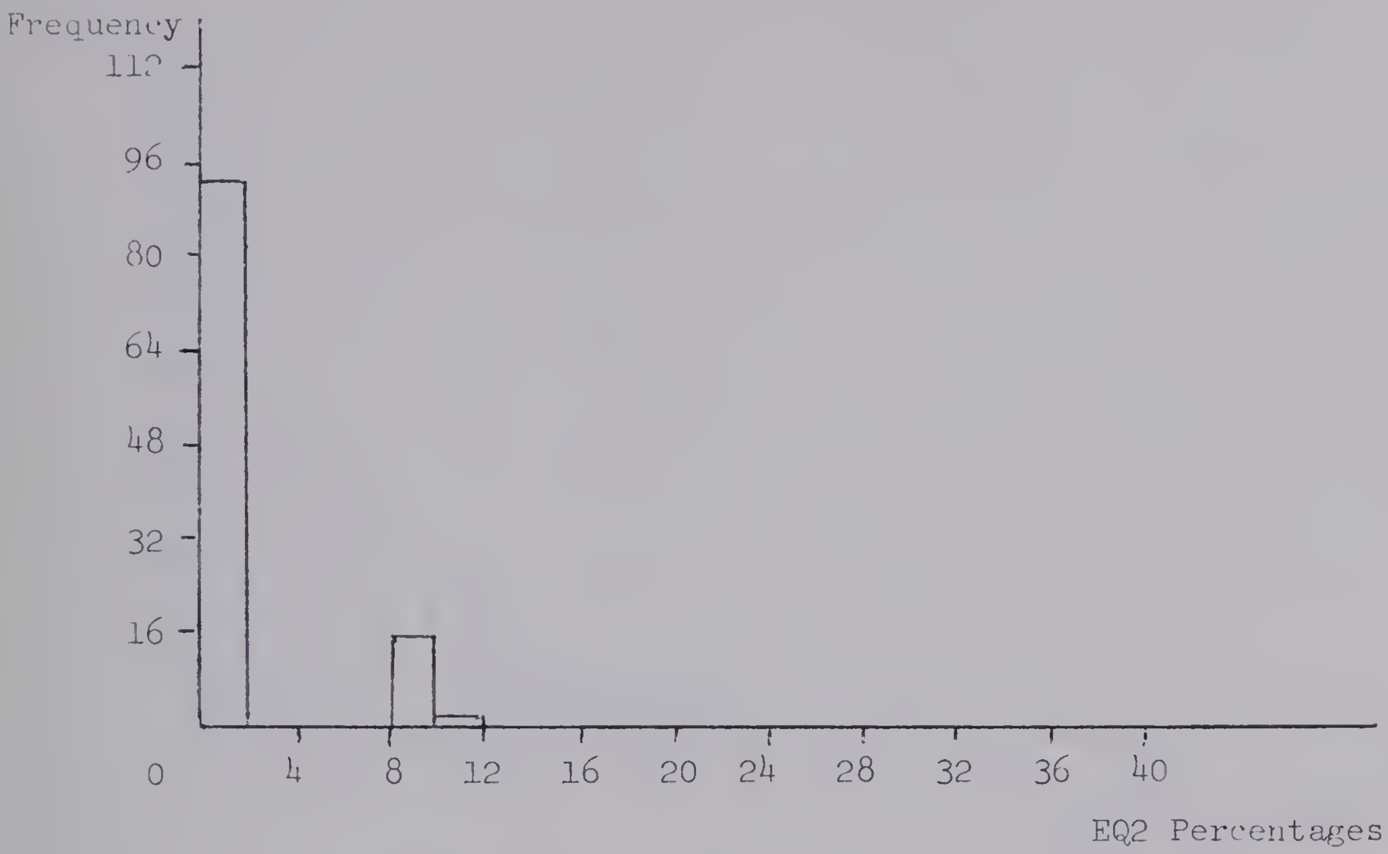

Fig. H12. Histogram of EQ2

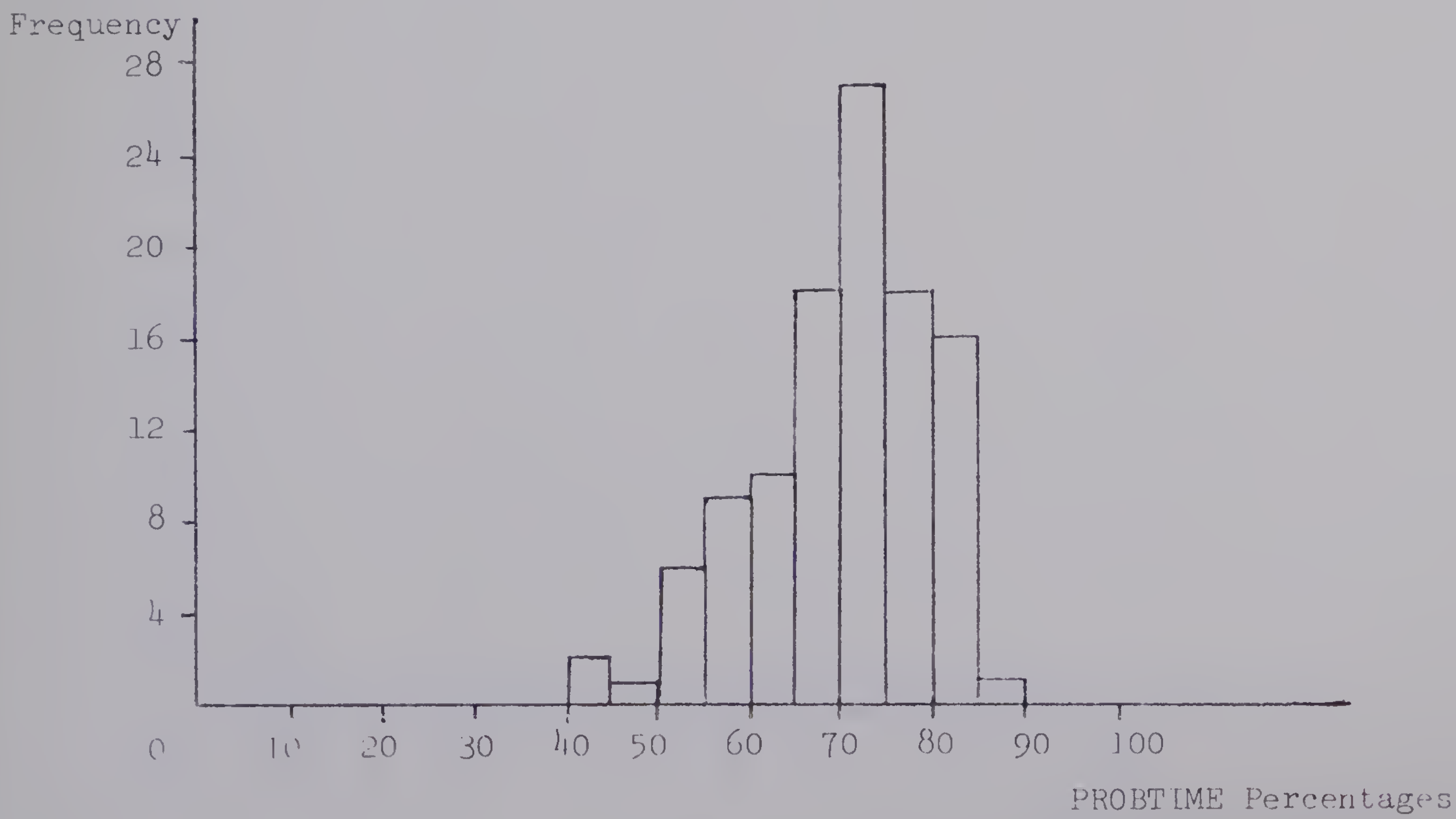

Fig. H13. Histogram of PROBTIME

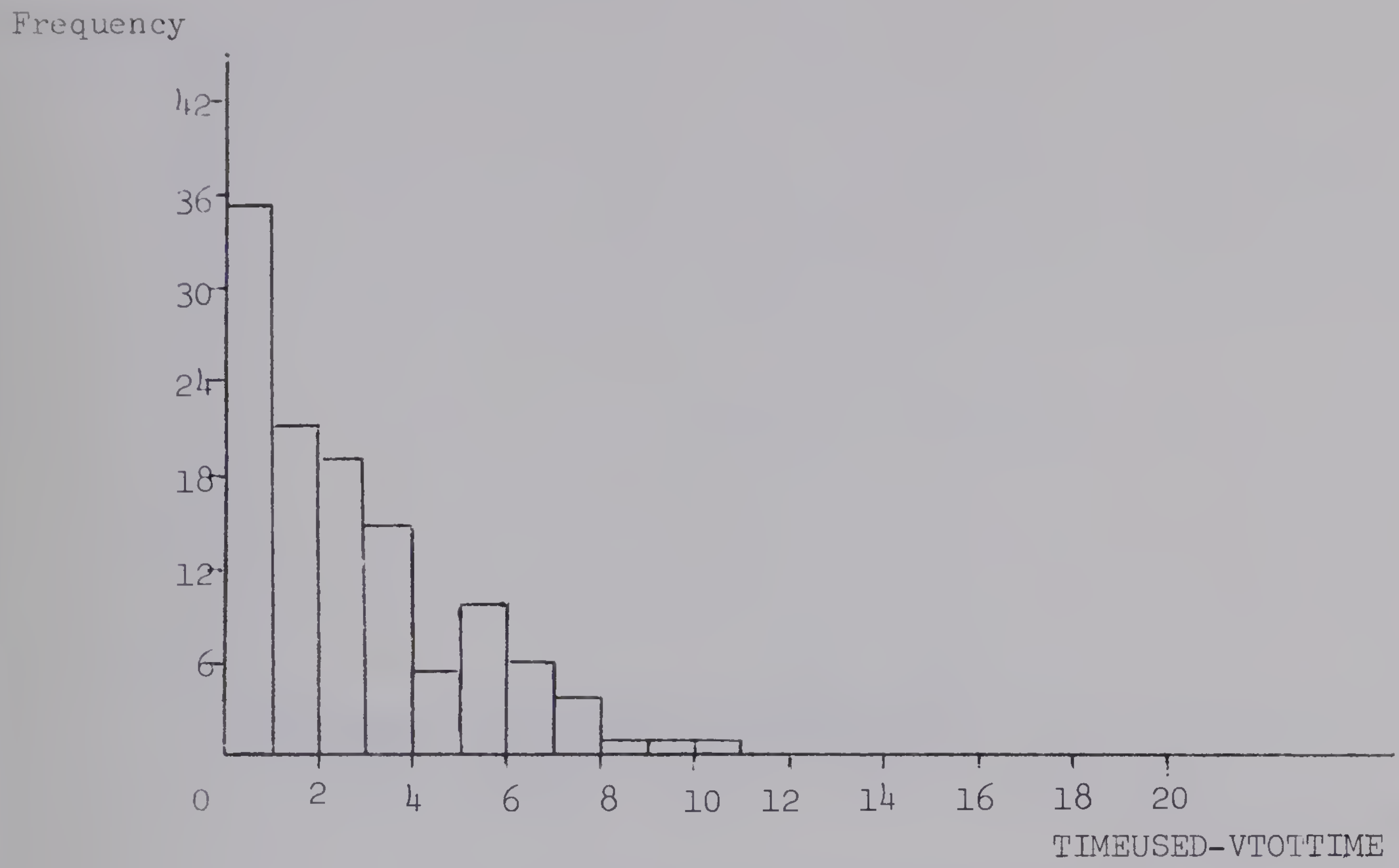

Fig. H14. Histogram of (TIMEUSED-VTOTTIME)

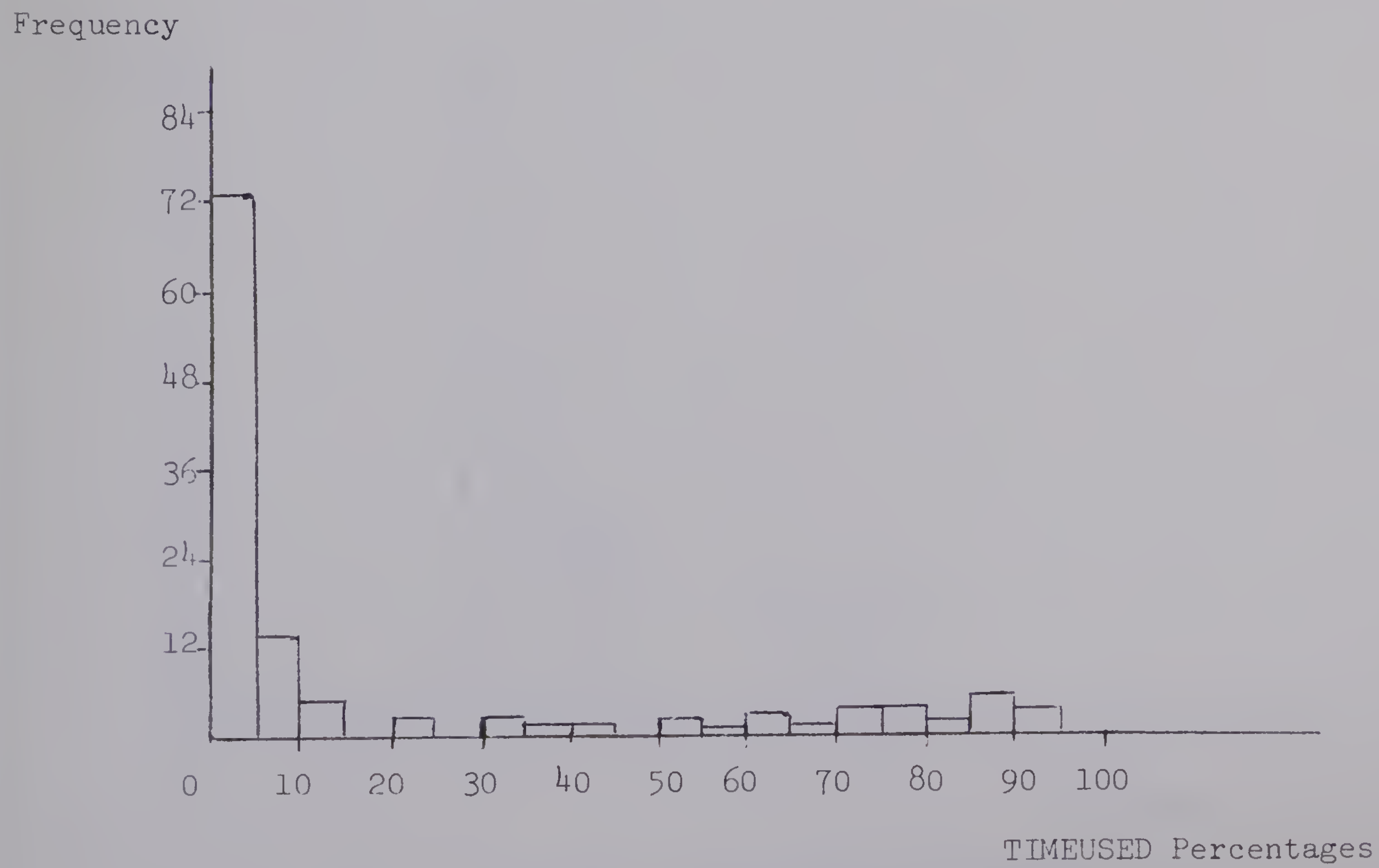

Fig. H15. Histogram of TIMEUSED

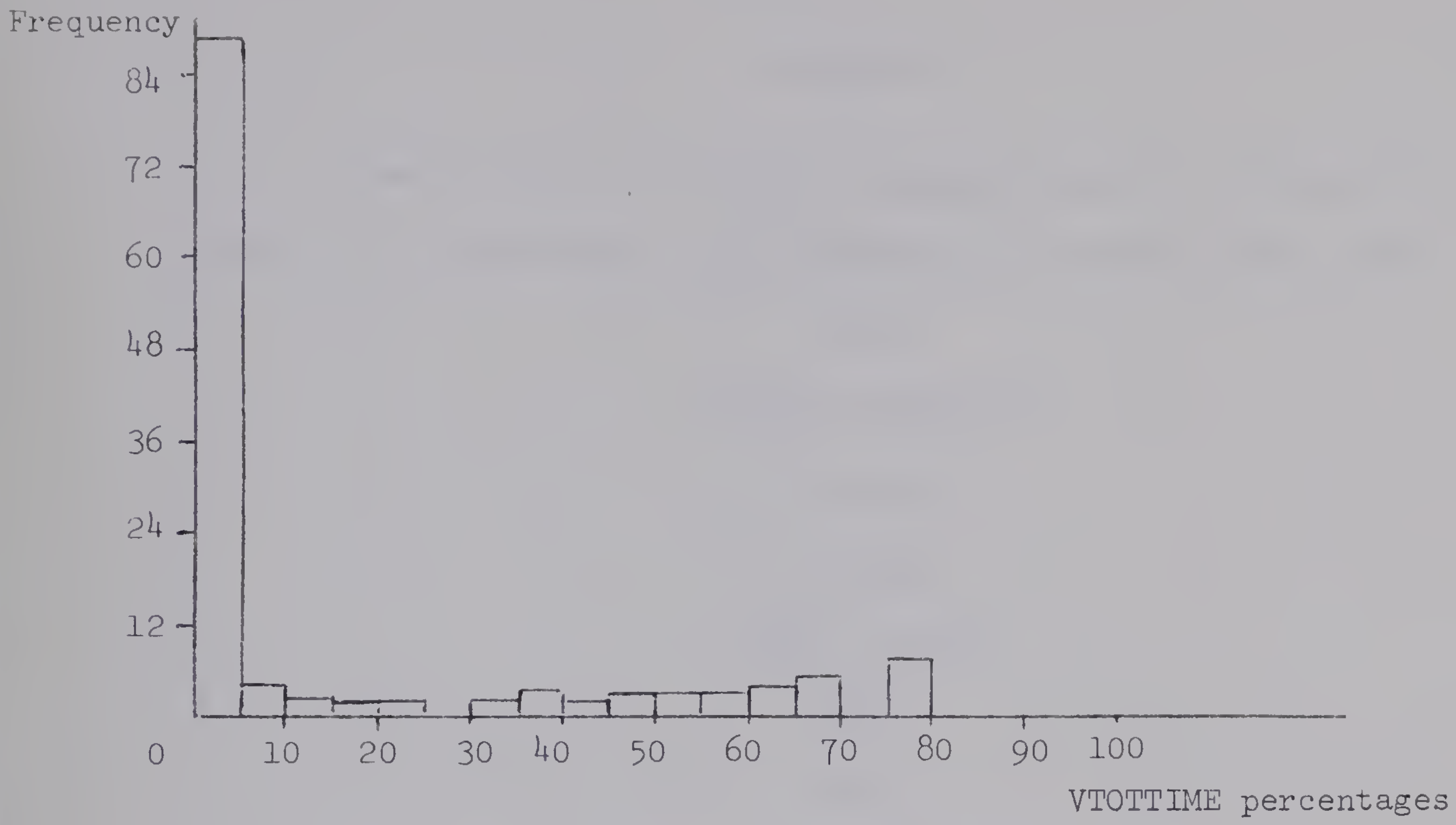

Fig. H16. Histogram of VTOTTIME

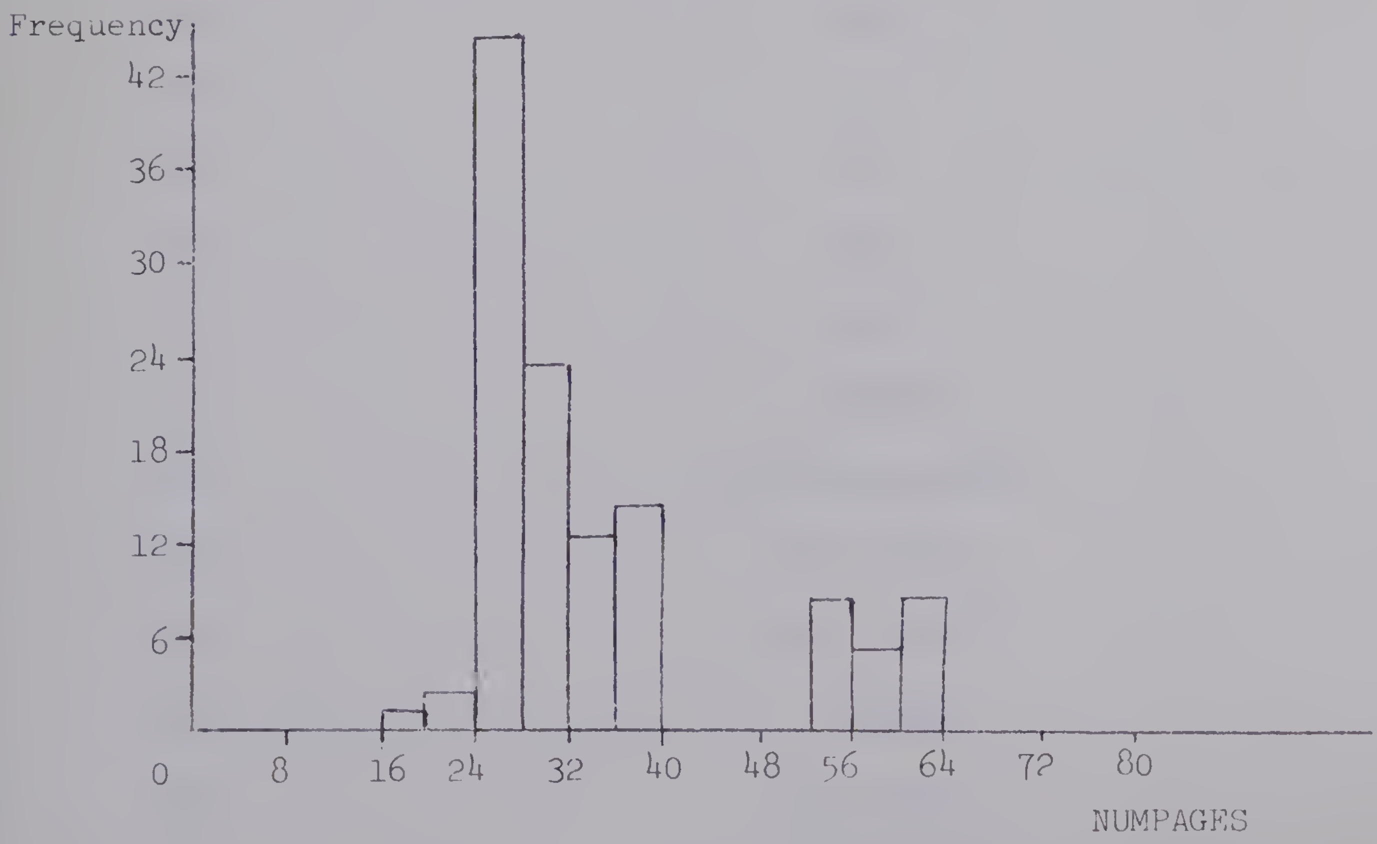

Fig. H17. Histogram of NUMPAGES

APPENDIX E

A sample output of the four regression results discussed in Section 5.5 is given here. The following 32 variables were used.

1	PGRD
2	$(\text{CPTIME-OVERHEAD})^2$
3	WAITTIME
4	OVERHEAD
5	(PGRD + PGSWP)
6	PGSWP
7	PGWT
8	IOWT
9	CFWT
10	EXCFN
11	Q2
12	Q1
13	EQ2
14	EQ1
15	PROBTIME
16	(CPTIME-OVERHEAD)
17	(EQ1-CFWT)
18	$(\text{EQ1-CFWT})^2$
19	$(\text{CPTIME})^2$
20	$(\text{WAITTIME})^2$
21	$(\text{OVERHEAD})^2$
22	$(\text{PGRD})^2$

23	$(PGSWP)^2$
24	$(PGWT)^2$
25	$(IOWT)^2$
26	$(CFWT)^2$
27	$(EXCFN)^2$
28	$(Q2)^2$
29	$(Q1)^2$
30	$(EQ2)^2$
31	$(EQ1)^2$
32	$(PROBTIME)^2$

The four dependent variables chosen were PROBTIME, (CPTIME - OVERHEAD), PGRD, and (PGRD + PGSWP). The first result is from Sample 7, with 1108 observations, and the remaining three results are from Sample 13 with 1117 observations. In all cases one variable was chosen as the dependent variable and the rest were considered as the independent variables. In some obvious cases, some variables were deleted before the regression analysis was carried out. For example, when (PGRD + PGSWP) was the dependent variable, PGRD and PGSWP were both deleted. An independent variable was allowed to enter the regression only if the reduction in the error sum of squares attributed to that variable was greater than one per cent of the total sum of squares.

```
i)  DEPENDENT VARIABLE . . . . . . . . . 15

    STEP  1

    VARIABLE ENTERED . . . . . 11

    SUM OF SQUARES REDUCED IN THIS STEP . . . .     353978.187
    PROPORTION REDUCED IN THIS STEP . . . . . .          0.285

    CUMULATIVE SUM OF SQUARES REDUCED . . . . .     353978.187
    CUMULATIVE PROPORTION REDUCED . . . . . . .          0.285  OF  1244109.000

    FOR  1 VARIABLES ENTERED
      MULTIPLE CORRELATION COEFFICIENT. . .     0.533
      F-VALUE FOR ANALYSIS OF VARIANCE. . .   439.823
      STANDARD ERROR OF ESTIMATE. . . . . .    28.369

      VARIABLE       REGRESSION       STD. ERROR OF       COMPUTED
       NUMBER        COEFFICIENT       REG. COEFF.        T-VALUE
        11             0.97233           0.04636           20.972
      INTERCEPT       47.62328

    STEP  2

    VARIABLE ENTERED . . . . . 13

    SUM OF SQUARES REDUCED IN THIS STEP . . . .     229267.125
    PROPORTION REDUCED IN THIS STEP . . . . . .          0.184

    CUMULATIVE SUM OF SQUARES REDUCED . . . . .     583245.312
    CUMULATIVE PROPORTION REDUCED . . . . . . .          0.469  OF  1244109.000

    FOR   2  VARIABLES ENTERED
      MULTIPLE CORRELATION COEFFICIENT. . .     0.685
      F-VALUE FOR ANALYSIS OF VARIANCE. . .   487.609
      STANDARD ERROR OF ESTIMATE. . . . . .    24.455
```

VARIABLE NUMBER	REGRESSION COEFFICIENT	STD. ERROR OF REG. COEFF.	COMPUTED T-VALUE
11	1.40948	0.04578	30.788
13	1.27821	0.06528	19.579
INTERCEPT	31.26590		

STEP 3

VARIABLE ENTERED 28

SUM OF SQUARES REDUCED IN THIS STEP 272214.437
PROPORTION REDUCED IN THIS STEP 0.219

CUMULATIVE SUM OF SQUARES REDUCED 855459.750
CUMULATIVE PROPORTION REDUCED . . . . 0.688 OF 1244109.000

FOR 3 VARIABLES ENTERED
MULTIPLE CORRELATION COEFFICIENT. . . 0.829
F-VALUE FOR ANALYSIS OF VARIANCE. . . 810.008
STANDARD ERROR OF ESTIMATE. 18.763

VARIABLE NUMBER	REGRESSION COEFFICIENT	STD. ERROR OF REG. COEFF.	COMPUTED T-VALUE
11	4.09521	0.10277	39.848
13	1.72998	0.05266	32.854
28	-0.05525	0.00199	-27.807
INTERCEPT	15.35425		

STEP 4

VARIABLE ENTERED 30

SUM OF SQUARES REDUCED IN THIS STEP 69302.125
PROPORTION REDUCED IN THIS STEP 0.056

CUMULATIVE SUM OF SQUARES REDUCED 924761.875

CUMULATIVE PROPORTION REDUCED . . . . 0.743 OF 1244109.000

FOR 4 VARIABLES ENTERED

MULTIPLE CORRELATION COEFFICIENT. . . 0.862

F-VALUE FOR ANALYSIS OF VARIANCE. . . 798.514

STANDARD ERROR OF ESTIMATE. 17.015

VARIABLE NUMBER	REGRESSION COEFFICIENT	STD. ERROR OF REG. COEFF.	COMPUTED T-VALUE
11	4.37991	0.09500	46.104
13	4.00746	0.15476	25.895
28	-0.05893	0.00182	-32.424
30	-0.05537	0.00358	-15.471
INTERCEPT	10.42419		

ANALYSIS OF VARIANCE FOR THE REGRESSION:

SOURCE OF VARIATION	DEGREES OF FREEDOM	SUM OF SQUARES	MEAN SQUARES	OVERALL F
ATTRIBUTABLE TO REGRESSION	4	924761.875	231190.469	798.514
DEVIATION FROM REGRESSION	1104	319337.125	289.255	
TOTAL	1108	1244109.000		

ii) DEPENDENT VARIABLE 16

STEP 1

VARIABLE ENTERED 5

SUM OF SQUARES REDUCED IN THIS STEP 2324.234
PROPORTION REDUCED IN THIS STEP 0.179

CUMULATIVE SUM OF SQUARES REDUCED 2324.234
CUMULATIVE PROPORTION REDUCED . . . . 0.179 OF 12978.723

FOR 1 VARIABLES ENTERED
MULTIPLE CORRELATION COEFFICIENT. . . 0.423
F-VALUE FOR ANALYSIS OF VARIANCE. . . 243.233
STANDARD ERROR OF ESTIMATE. 3.091

VARIABLE NUMBER	REGRESSION COEFFICIENT	STD. ERROR OF REG. COEFF.	COMPUTED T-VALUE
5	0.53733	0.03445	15.596
INTERCEPT	4.03443		

STEP 2

VARIABLE ENTERED 8

SUM OF SQUARES REDUCED IN THIS STEP 858.608
PROPORTION REDUCED IN THIS STEP 0.066

CUMULATIVE SUM OF SQUARES REDUCED 3182.843
CUMULATIVE PROPORTION REDUCED . . . . 0.245 OF 12978.723

FOR 2 VARIABLES ENTERED
MULTIPLE CORRELATION COEFFICIENT. . . 0.495
F-VALUE FOR ANALYSIS OF VARIANCE. . . 180.978
STANDARD ERROR OF ESTIMATE. 2.965

VARIABLE NUMBER	REGRESSION COEFFICIENT	STD. ERROR OF REG. COEFF.	COMPUTED T-VALUE
5	0.48001	0.03356	14.305
8	0.30462	0.03083	9.881
INTERCEPT	3.94738		

STEP 3

VARIABLE ENTERED 9

SUM OF SQUARES REDUCED IN THIS STEP 256.054
PROPORTION REDUCED IN THIS STEP 0.020

CUMULATIVE SUM OF SQUARES REDUCED 3438.897
CUMULATIVE PROPORTION REDUCED . . . . 0.265 OF 12978.723

FOR 3 VARIABLES ENTERED
MULTIPLE CORRELATION COEFFICIENT. . . 0.515
F-VALUE FOR ANALYSIS OF VARIANCE. . . 133.737
STANDARD ERROR OF ESTIMATE. 2.928

VARIABLE NUMBER	REGRESSION COEFFICIENT	STD. ERROR OF REG. COEFF.	COMPUTED T-VALUE
5	0.47637	0.03314	14.376
8	0.29584	0.03048	9.706
9	-0.07727	0.01414	-5.466
INTERCEPT	4.13531		

ANALYSIS OF VARIANCE FOR THE REGRESSION:

SOURCE OF VARIATION	DEGREES OF FREEDOM	SUM OF SQUARES	MEAN SQUARES	OVERALL F
ATTRIBUTABLE TO REGRESSION	3	3438.897	1146.299	133.737
DEVIATION FROM REGRESSION	1114	9539.826	8.565	
TOTAL	1117	12978.723		

iii) DEPENDENT VARIABLE 1

STEP 1

VARIABLE ENTERED 6

SUM OF SQUARES REDUCED IN THIS STEP 2166.911
PROPORTION REDUCED IN THIS STEP 0.958

CUMULATIVE SUM OF SQUARES REDUCED 2166.911
CUMULATIVE PROPORTION REDUCED . . . . 0.958 OF 2261.827

FOR 1 VARIABLES ENTERED
MULTIPLE CORRELATION COEFFICIENT. . . 0.979
F-VALUE FOR ANALYSIS OF VARIANCE. . . 25455.195
STANDARD ERROR OF ESTIMATE. 0.292

VARIABLE NUMBER	REGRESSION COEFFICIENT	STD. ERROR OF REG. COEFF.	COMPUTED T-VALUE
6	1.09167	0.00684	159.547
INTERCEPT	0.01632		

ANALYSIS OF VARIANCE FOR THE REGRESSION:

SOURCE OF VARIATION	DEGREES OF FREEDOM	SUM OF SQUARES	MEAN SQUARES	OVERALL F
ATTRIBUTABLE TO REGRESSION	1	2166.911	2166.911	25455.195
DEVIATION FROM REGRESSION	1116	94.916	0.085	
TOTAL	1117	2261.827		

iv) DEPENDENT VARIABLE 5

STEP 1

VARIABLE ENTERED 16

```
SUM OF SQUARES REDUCED IN THIS STEP . . . .      1441.625
PROPORTION REDUCED IN THIS STEP . . . . . .         0.179

CUMULATIVE SUM OF SQUARES REDUCED . . . . .      1441.625
CUMULATIVE PROPORTION REDUCED . . . . . . .         0.179  OF  8050.156

FOR  1  VARIABLES ENTERED
  MULTIPLE CORRELATION COEFFICIENT. . .     0.423
  F-VALUE FOR ANALYSIS OF VARIANCE. . .   243.233
  STANDARD ERROR OF ESTIMATE. . . . . .     2.435
```

VARIABLE NUMBER	REGRESSION COEFFICIENT	STD. ERROR OF REG. COEFF.	COMPUTED T-VALUE
16	0.33328	0.02137	15.596
INTERCEPT	-0.66428		

STEP 2

VARIABLE ENTERED 29

```
SUM OF SQUARES REDUCED IN THIS STEP . . . .       588.910
PROPORTION REDUCED IN THIS STEP . . . . . .         0.073

CUMULATIVE SUM OF SQUARES REDUCED . . . . .      2030.535
CUMULATIVE PROPORTION REDUCED . . . . . . .         0.252  OF  8050.156

FOR  2  VARIABLES ENTERED
  MULTIPLE CORRELATION COEFFICIENT. . .     0.502
  F-VALUE FOR ANALYSIS OF VARIANCE. . .   187.887
  STANDARD ERROR OF ESTIMATE. . . . . .     2.325
```

VARIABLE NUMBER	REGRESSION COEFFICIENT	STD. ERROR OF REG. COEFF.	COMPUTED T-VALUE
16	0.30368	0.02060	14.742
29	0.00319	0.00031	10.440
INTERCEPT	-0.94698		

STEP 3

VARIABLE ENTERED 21

SUM OF SQUARES REDUCED IN THIS STEP 222.480
PROPORTION REDUCED IN THIS STEP 0.028

CUMULATIVE SUM OF SQUARES REDUCED 2253.015
CUMULATIVE PROPORTION REDUCED . . . . 0.280 OF 8050.156

FOR 3 VARIABLES ENTERED
MULTIPLE CORRELATION COEFFICIENT. . . 0.529
F-VALUE FOR ANALYSIS OF VARIANCE. . . 144.186
STANDARD ERROR OF ESTIMATE. 2.282

VARIABLE NUMBER	REGRESSION COEFFICIENT	STD. ERROR OF REG. COEFF.	COMPUTED T-VALUE
16	0.42903	0.02787	15.392
29	0.00262	0.00031	8.393
21	-0.03839	0.00587	-6.536
INTERCEPT	-1.03950		

STEP 4

VARIABLE ENTERED 7

SUM OF SQUARES REDUCED IN THIS STEP 170.127
PROPORTION REDUCED IN THIS STEP 0.021

CUMULATIVE SUM OF SQUARES REDUCED 2423.141
CUMULATIVE PROPORTION REDUCED . . . . 0.301 OF 8050.156

FOR 4 VARIABLES ENTERED
MULTIPLE CORRELATION COEFFICIENT. . . 0.549
F-VALUE FOR ANALYSIS OF VARIANCE. . . 119.714
STANDARD ERROR OF ESTIMATE. 2.250

VARIABLE NUMBER	REGRESSION COEFFICIENT	STD. ERROR OF REG. COEFF.	COMPUTED T-VALUE
16	0.41511	0.02758	15.052
29	0.00199	0.00033	6.079
21	-0.03809	0.00579	-6.579
7	0.14772	0.02548	5.798
INTERCEPT	-0.95573		

ANALYSIS OF VARIANCE FOR THE REGRESSION:

SOURCE OF VARIATION	DEGREES OF FREEDOM	SUM OF SQUARES	MEAN SQUARES	OVERALL F
ATTRIBUTABLE TO REGRESSION	4	2423.141	605.785	119.714
DEVIATION FROM REGRESSION	1113	5627.015	5.056	
TOTAL	1117	8050.156		